

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of : THE COMMISSIONER IS AUTHORIZED
Kohsaku SHIBATA : TO CHARGE ANY DEFICIENCY IN THE
Serial No. NEW : FEES FOR THIS PAPER TO DEPOSIT
Filed December 9, 2003 : ACCOUNT NO. 23-0975
: **Attn: APPLICATION BRANCH**
: Attorney Docket No. 2003_1791A

SIMULATION APPARATUS, METHOD
AND PROGRAM

CLAIM OF PRIORITY UNDER 35 USC 119

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450


Sir:

Applicant in the above-entitled application hereby claims the date of priority under the International Convention of Japanese Patent Application No. 2002-360362, filed December 12, 2002, as acknowledged in the Declaration of this application.

A certified copy of said Japanese Patent Application is submitted herewith.

Respectfully submitted,

Kohsaku SHIBATA

By 
Michael S. Huppert
Registration No. 40,268
Attorney for Applicant

MSH/kjf
Washington, D.C. 20006-1021
Telephone (202) 721-8200
Facsimile (202) 721-8250
December 9, 2003

日 本 国 特 許 庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出 願 年 月 日

Date of Application:

2002年12月12日

出 願 番 号

Application Number:

特願2002-360362

[ST.10/C]:

[JP2002-360362]

出 願 人

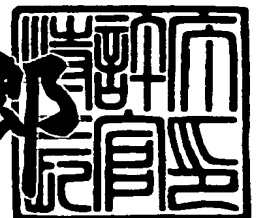
Applicant(s):

松下電器産業株式会社

2003年 5月20日

特 許 庁 長 官
Commissioner,
Japan Patent Office

太田 信一郎



出証番号 出証特2003-3037488

【書類名】 特許願

【整理番号】 5037740114

【あて先】 特許庁長官殿

【国際特許分類】 G06F

【発明者】

 【住所又は居所】 大阪府門真市大字門真 1 0 0 6 番地 松下電器産業株式会社内

 【氏名】 柴田 耕作

【特許出願人】

 【識別番号】 000005821

 【氏名又は名称】 松下電器産業株式会社

【代理人】

 【識別番号】 100109210

 【弁理士】

 【氏名又は名称】 新居 広守

 【電話番号】 06-4806-7530

【手数料の表示】

 【予納台帳番号】 049515

 【納付金額】 21,000円

【提出物件の目録】

 【物件名】 明細書 1

 【物件名】 図面 1

 【物件名】 要約書 1

 【包括委任状番号】 0213583

【ブルーフの要否】 要

【書類名】 明細書

【発明の名称】 シミュレーション装置、シミュレーション方法及びプログラム

【特許請求の範囲】

【請求項 1】 シミュレーション対象の V L I W プロセッサによって同時実行されるべき複数命令に対して、当該複数命令の逐次実行に対応する命令単位のシミュレーション結果を生成する

ことを特徴とするシミュレーション装置。

【請求項 2】 前記シミュレーション装置は、同時実行されるべき複数命令からなる命令群をシミュレーションする第 1 シミュレーション手段と、

第 1 シミュレーション手段によるシミュレーション結果に基づいて、前記命令群中の命令毎のシミュレーション結果を生成する第 2 シミュレーション手段とを備えることを特徴とする請求項 1 記載のシミュレーション装置。

【請求項 3】 前記第 1 シミュレーション手段は、V L I W プロセッサのリソースデータを示す第 1 データを記憶する記憶手段と第 1 データのコピーを第 2 データとして記憶手段に保存する保存手段と、保存後に、1 つの命令群の実行をシミュレーションすることにより第 1 データを更新する第 1 シミュレータとを備え、

前記第 2 シミュレーション手段は、第 1 データおよび第 2 データに基づいて、当該命令群中の命令について命令単位のシミュレーション結果を得ることを特徴とする請求項 2 記載のシミュレーション装置。

【請求項 4】 前記保存手段は、レジスタセットのデータを第 2 データとして記憶手段に保存し、

前記第 2 シミュレーション手段は、前記命令群中の命令単位に当該命令のシミュレーション実行前の状態にリソースを復元する

ことを特徴とする請求項 3 記載のシミュレーション装置。

【請求項 5】 前記保存手段は、さらに、前記命令群にメモリライト命令が存在する場合、ライト前のメモリデータを第 2 データに含ませて記憶手段に保存する

ことを特徴とする請求項 4 記載のシミュレーション装置。

【請求項 6】 前記第 2 シミュレーション手段は、
ブレイク条件を満たす命令が第 1 シミュレーション手段によりシミュレーションされた命令群に含まれているか否か判定する判定手段と、
含まれていないと判定された場合に、第 1 シミュレーション手段に次の命令群のシミュレーションを指示する指示手段と、
含まれていると判定された場合に、前記ブレイク条件を満たす命令を停止命令と決定する決定手段と

を備えることを特徴とする請求項 5 記載のシミュレーション装置。

【請求項 7】 前記判定手段は、命令単位のシミュレーションのステップ実行において、現在の停止命令の次の命令をブレイク条件とする
ことを特徴とする請求項 6 記載のシミュレーション装置。

【請求項 8】 前記第 2 シミュレーション手段は、さらに、
前記第 1 データ及び第 2 データに基づいて、決定手段に決定された停止命令の直前の命令までシミュレーションした場合のリソースデータを復元する復元手段を備える

ことを特徴とする請求項 6 記載のシミュレーション装置。

【請求項 9】 前記第 2 シミュレーション手段は、さらに、
前記第 1 データ及び第 2 データに基づいて、決定手段に決定された停止命令までシミュレーションした場合のリソースデータを復元する復元手段を備える
ことを特徴とする請求項 6 記載のシミュレーション装置。

【請求項 10】 前記第 1 シミュレータは、前記命令群中の各命令について当該命令により変更されるリソースを示す更新情報を生成し、

前記復元手段は、第 1 データ、第 2 データ及び更新情報に従って当該命令群中の命令毎に当該命令までの逐次実行した結果に対応するリソースデータを復元する

ことを特徴とする請求項 8 又は 9 記載のシミュレーション装置。

【請求項 11】 前記第 1 シミュレータは、パイプライン処理を行う VLIW プロセッサを対象として、パイプライン処理のサイクル単位で命令群のシミュレーションを行い、

前記シミュレーション装置は、さらに、命令群単位のシミュレーションにおける実行サイクル数をカウントする

ことを特徴とする請求項 3 記載のシミュレーション装置。

【請求項 12】 前記 VLIW プロセッサは、同時実行されるべき複数命令中の命令実行をキャンセルするキャンセル機構を有し、

前記第 1 シミュレータは、キャンセル機構をシミュレーションすることを特徴とする請求項 11 記載のシミュレーション装置。

【請求項 13】 前記第 1 シミュレータは、さらに、シミュレーション対象の VLIW プロセッサの実行ステージで遅延サイクルを発生させる遅延命令について、遅延サイクルをシミュレーションし、

前記復元手段は、遅延命令についての更新情報に従って遅延命令の実行結果に対応するリソースデータを生成する

ことを特徴とする請求項 11 記載のシミュレーション装置。

【請求項 14】 前記復元手段は、さらに、シミュレーション対象の VLIW プロセッサの実行ステージで遅延サイクルを発生させる遅延命令との間で、同一命令群内で出力依存関係を有する出力依存命令について、遅延命令についての更新情報と出力依存命令についての更新情報に従って、出力依存命令の実行結果に対応するリソースデータを生成する

ことを特徴とする請求項 13 記載のシミュレーション装置。

【請求項 15】 シミュレーション対象の VLIW プロセッサによって同時実行されるべき複数命令に対して、当該複数命令の逐次実行に対応する命令単位のシミュレーション結果を生成する

ことを特徴とするシミュレーション方法。

【請求項 16】 前記シミュレーション方法は、

同時実行されるべき複数命令からなる命令群をシミュレーションする第 1 ステ

ップと、

第 1 ステップにおけるシミュレーション結果に基づいて、前記命令群中の命令毎のシミュレーション結果を生成する第 2 ステップと

を有することを特徴とする請求項 1 5 記載のシミュレーション方法。

【請求項 1 7】 前記第 1 ステップは、

V L I W プロセッサの現在のリソース内容を第 1 データのコピーを第 2 データとして記憶部に保存する保存サブステップと、

保存後に、前記命令群の実行をシミュレーションすることにより第 1 データを更新する更新ステップと

を有し、

前記第 2 ステップにおいて、第 1 データおよび第 2 データに基づいて、前記命令群中の命令単位のシミュレーション結果を得る

ことを特徴とする請求項 1 6 記載のシミュレーション方法。

【請求項 1 8】 前記保存サブステップにおいて、レジスタセットのデータを第 2 データとして記憶部に保存し、

前記第 2 ステップにおいて、前記命令群中の命令単位の当該命令のシミュレーション実行前の状態にリソースを復元する

ことを特徴とする請求項 1 7 記載のシミュレーション方法。

【請求項 1 9】 前記第 2 ステップにおいて、

ブレイク条件を満たす命令が第 1 ステップにおいてシミュレーションされた命令群に含まれているか否か判定する判定サブステップと、

含まれていないと判定された場合に、第 1 ステップにおいて次の命令群のシミュレーションを再度行わせ、含まれていると判定された場合に、前記ブレイク条件を満たす命令を停止命令と決定する決定サブステップと

を有することを特徴とする請求項 1 8 記載のシミュレーション方法。

【請求項 2 0】 前記更新ステップは、パイプライン処理を行う V L I W プロセッサを対象として、パイプライン処理のサイクル単位で命令群のシミュレーションを行い、

前記シミュレーション方法は、さらに、命令群単位のシミュレーションにおけ

る実行サイクル数をカウントする

ことを特徴とする請求項 1 9 記載のシミュレーション方法。

【請求項 2 1】 請求項 1 5 ないし 2 0 の何れか 1 項に記載のシミュレーション方法をコンピュータに実行させることを特徴とするプログラム。

【発明の詳細な説明】

【 0 0 0 1 】

【発明の属する技術分野】

V L I W (Very Long Instruction Word) プロセッサ用のプログラムをシミュレーションすることによりソフトウェア開発者のプログラム開発を支援するシミュレーション装置に関する。

【 0 0 0 2 】

【従来の技術】

プロセッサの実行状態を模倣するシミュレーション装置は、プログラム開発において有用である。パイプライン制御を行うプロセッサを対象とするシミュレーション装置では、パイプラインのシミュレーションを正確に行うため、サイクル単位でのシミュレーションを行っている。これに関する先行技術文献として特許文献 1 がある。

【 0 0 0 3 】

この特許文献 1 に開示されたパイプライン計算機シミュレータは、パイプラインのシミュレーションを行いながら 1 命令単位でのステップ動作のシミュレーションをおこなう。これにより、デバッグ動作に有用な 1 命令のステップ動作が直感的に正しく動作する。

【 0 0 0 4 】

【特許文献 1】

特開平 8 - 2 7 2 6 1 2

【 0 0 0 5 】

【発明が解決しようとする課題】

しかしながら、上記従来技術では、単一命令発行のプロセッサを対象として命令単位のステップ動作を行うものの、複数命令を同時に実行するプロセッサを対

象とするシミュレーション装置は、複数命令の同時実行をシミュレーションすることから、ソフトウェア開発者は命令単位の実行過程まで細かく確認することができなかった。

【 0 0 0 6 】

本発明は、複数命令を同時実行するプロセッサを対象とし、命令単位の実行過程がわかるようにシミュレーションを行うシミュレーション装置を提供することを目的とする。

【 0 0 0 7 】

【課題を解決するための手段】

上記課題を解決するため本発明のシミュレーション装置は、シミュレーション対象の V L I W プロセッサによって同時実行されるべき複数命令に対して、当該複数命令の逐次実行に対応する命令単位のシミュレーション結果を生成するように構成されている。

【 0 0 0 8 】

この構成によれば、複数命令を同時実行するプロセッサにおける命令実行動作を、命令単位の実行過程まできめ細かくデバッグ及び検証するのに役立つ。例えば、同時に実行される命令を個別に 1 命令ずつデバッグすることができる。

【 0 0 0 9 】

また、前記シミュレーション装置は、同時実行されるべき複数命令からなる命令群をシミュレーションする第 1 シミュレーション手段と、第 1 シミュレーション手段によるシミュレーション結果に基づいて、前記命令群中の命令毎のシミュレーション結果を生成する第 2 シミュレーション手段とを備える構成としてもよい。

【 0 0 1 0 】

この構成によれば、命令群のシミュレーション結果を利用して、命令群中の個々の命令についてシミュレーション結果を生成可能であり、ソフトウェア開発者に対しては、あたかも命令単位に逐次シミュレーションしているように見せることができるので、命令個別のデバッグ及び検証を容易にすることができる。

【 0 0 1 1 】

さらに、前記第1シミュレーション手段は、VLIWプロセッサの現在のリソースを第1データとして記憶する記憶部と、第1データのコピーを第2データとして記憶手段に保存する保存手段と、保存後に、前記命令群の実行をシミュレーションすることにより第1データを更新する第1シミュレータとを備え、前記第2シミュレーション手段は、第1データおよび第2データに基づいて、前記命令群中の命令単位のシミュレーションを結果を得る構成としてもよい。

【0012】

この構成によれば、1つの命令群のシミュレーション前のリソースデータを示す第2データとシミュレーション後のリソースデータを示す第1データとに基づいて、命令単位のシミュレーションを結果を得ている。

【0013】

また、前記第2シミュレーション手段は、ブレイク条件を満たす命令が第1シミュレーション手段によりシミュレーションされた命令群に含まれているか否か判定する判定手段と、含まれていないと判定された場合に、第1シミュレーション手段に次の命令群のシミュレーションを指示する指示手段と、含まれていると判定された場合に、前記ブレイク条件を満たす命令を停止命令と決定する決定手段とを備える構成としてもよい。

【0014】

この構成によれば、同時実行される複数命令に対して命令個別にブレイクさせることが可能なので、ソフトウェア開発者にとって、同時実行される複数命令における命令の相互関係を容易に確認することができる。また、シミュレーション済の命令群に含まれる命令を対象にブレイク条件を判定しているので、プロセッサにおける実際の実行経路においてブレイクさせることができる。例えば、命令群中にキャンセルされた命令がある場合には、実行されない命令を誤ってブレイクさせることがない。

【0015】

ここで、前記第1シミュレータは、パイプライン処理を行うVLIWプロセッサを対象として、パイプライン処理のサイクル単位で命令群のシミュレーションを行い、前記シミュレーション装置は、さらに、命令群単位のシミュレーション

における実行サイクル数をカウントする構成としてもよい。

【0016】

この構成によれば、パイプライン処理するプロセッサを対象に、命令単位のシミュレーション結果を生成しながらも、プロセッサのパイプラインサイクル数を正確にシミュレーションすることができる。

【0017】

また、前記第1シミュレータは、前記命令群中の各命令について当該命令により変更されるリソースを示す更新情報を生成し、前記復元手段は、第1データ、第2データ及び更新情報に従って当該命令群中の命令毎に当該命令までの逐次実行した結果に対応するリソースデータを復元する構成としてもよい。

【0018】

この構成によれば、更新情報を利用することによりリソースデータの復元を容易にすることができる。

ここで、前記第1シミュレータは、さらに、VLIWプロセッサの実行ステージで遅延サイクルを発生させる遅延命令について、遅延サイクルをシミュレーションし、前記復元手段は、遅延命令についての更新情報に従って遅延命令の実行結果に対応するリソースデータを生成する構成としてもよい。

【0019】

この構成によれば、プロセッサにおいては、同時実行される複数命令中の遅延命令は遅れて実行されるので、前記複数命令はサイクルレベルでは同時ではなく実行順序が入れ替わるが、命令単位のシミュレーションにおいては1つずつ順に実行したシミュレーション結果を生成することができる。

【0020】

さらに、前記復元手段は、さらに、シミュレーション対象のVLIWプロセッサの実行ステージで遅延サイクルを発生させる遅延命令との間で、同一命令群内で出力依存関係を有する出力依存命令について、遅延命令についての更新情報と出力依存命令についての更新情報に従って、出力依存命令の実行結果に対応するリソースデータを生成する構成としてもよい。

【0021】

この構成によれば、プロセッサにおいては、同時実行される出力依存関係にある命令の一方の実行結果はキャンセルしたのと同じ結果になるが、命令単位のシミュレーションでは、出力依存関係によりキャンセルされる運命にある命令についてもシミュレーション結果を生成するので、ソフトウェア開発者は、結果的に出力依存命令が結果的にキャンセルされる過程を知ることができる。

【 0 0 2 2 】

【発明の実施の形態】

＜シミュレーションシステムの概要＞

図 1 は、本発明の実施の形態におけるシミュレーションシステム 1 の外観を示す図である。同図のシミュレーションシステム 1 は、シミュレーション装置 2 とデバッグ装置 3 とからなる。

【 0 0 2 3 】

シミュレーション装置 2 は、本体装置 2 a、表示装置 2 b、入力装置 2 c を備え、複数命令を同時に実行するプロセッサの動作をシミュレーションする装置であって、プロセッサでは同時実行される複数命令について、命令を一つずつ順に実行した場合に得られる実行結果を LAN ケーブルを介してデバッグ装置 3 に通知するよう構成されている。すなわちシミュレーション装置 2 は、ユーザからデバッグ装置 3 を介して命令単位のステップ動作指示や、命令単位のブレークポイント指示を受け付けて、命令単位の実行結果をデバッグ装置 3 を介してユーザに対して表示する。本体装置 2 a は、シミュレーションソフトウェアを実行することにより上記のシミュレーションを行う。表示装置 2 b、入力装置 2 c はシミュレーションソフトウェアの実行開始と終了に用いられる。

【 0 0 2 4 】

デバッグ装置 3 は、本体装置 3 a、表示装置 3 b、入力装置 3 c を備え、シミュレーション装置 2 に対するユーザインターフェースとして機能し、ユーザから、命令単位のステップ動作や、命令単位のブレークポイント指定などの操作を受けてシミュレーション装置 2 に通知し、シミュレーション装置 2 からの命令単位のシミュレーション結果を受信して表示する。本体装置 3 a は、デバッグソフトウェアを実行することにより、ユーザインターフェース機能を提供し、シミュレ

ーション装置 2 に対してシミュレーション実行指示やその応答を送受信する。表示装置 3 b は、命令単位のシミュレーション実行結果などを表示する。入力装置 3 c は各種ユーザ操作を受け付ける。

【 0 0 2 5 】

図 2 はシミュレーション装置 2 においてシミュレーションソフトウェア起動後の表示内容の一例を示す。同図において、W 0 はコンソールウィンドウであり、シミュレーションソフトウェアが起動されて実行中の状態を示している。この状態において、デバッグ装置 3 を介してソフトウェア開発者が本シミュレーション装置 2 を利用可能となる。

【 0 0 2 6 】

図 3 は、デバッグ装置 3 においてデバッグソフトウェア実行中の表示内容の一例を示す。同図において、W 1 はデバッグ対象のプログラムを表示するコード表示ウィンドウ、W 2 はデバッグ対象のソースプログラムを表示するソース表示ウィンドウ、W 3 はユーザ操作により各種シミュレーション用のコマンドを入力するためのコマンド入力ウィンドウ、W 4 はターゲットプロセッサにおいて実行した場合のサイクル数とシミュレーションした命令数を示すステップ数等を表示する状態表示ウィンドウ、W 5 はレジスタデータを表示するレジスタ内容表示ウィンドウ、W 6 はメモリデータを表示するメモリ内容表示ウィンドウである。M 1 は命令単位のシミュレーションにおけるシミュレーション未実行の先頭命令（以下、停止命令と呼ぶ）を示す停止命令マークであり、M 2 は停止命令に対応するソースプログラム中のプログラム行を示す停止行マークである。

【 0 0 2 7 】

コード表示ウィンドウ W 1 は、デバッグ対象のプログラムについて命令アドレスを示すプログラムカウント値（同図の P C 欄）、行番号（L I N E 欄）、フラグ（F L G 欄）、ニーモニック（M N E M O N I C 欄）等を表示するとともに、停止命令マーク M 1 を表示している。ここで、フラグ [F 0] は、V L I W プロセッサ 2 0 0 に備えられた状態フラグレジスタ中のフラグ F 0 の値により命令が有効か否かを示し、直前の比較命令（c m p 命令）における比較結果に応じてリセットされ得る。このフラグ F 0 は、条件分岐命令における条件の成否をフラグ

F0の値に反映させることにより、例えばソース表示ウィンドウW2中のif文の条件に依存する実行文などに利用される。またコード表示ウィンドウW1における;;（セミコロン2つ）は、ターゲットプロセッサにおいて同時実行される命令の切れ目を示し、例えばデータ依存関係のある2つの命令など同時実行できない直後の命令を次の命令群に分けるために使用される。

【0028】

なお、図1に示したシミュレーションシステム1は、シミュレーションソフトウェアを実行するコンピュータ（シミュレーション装置2）とデバッグソフトウェアを実行するコンピュータ（デバッガ3）の2台のコンピュータから構成されるが、1台のコンピュータによりシミュレーションソフトウェアとデバッグソフトウェアの両方を実行する構成としてもよい。

<ターゲットプロセッサ>

次に、シミュレーションシステム1のターゲットプロセッサの仕様について図4～図11を用いて説明する。

【0029】

図4は、シミュレーション対象のVLIWプロセッサの構成を示すブロック図である。同図のように、プロセッサ200は、IF（命令フェッチ）ステージ、DC（デコード）ステージ、EX/MEM（実行/メモリアクセス）ステージ、WB（ライトバック）ステージからなる4段パイプライン構成であり、IFステージにおいて複数命令からなる命令群をフェッチする命令フェッチ制御部201と、DCステージにおいてフェッチされた最大3個の命令を同時にデコードする命令デコーダ202～204と、EXステージにおいてデコード結果に従って最大3個の命令を同時に実行するALU（Arithmetic Logic Unit）205～207と、デコードされた命令がメモリアクセス命令である場合にMEMステージにおいてその命令を実行するメモリアクセス制御部209と、命令の実行内容あるいは一部の実行内容をキャンセルする命令キャンセル部220と、次段のステージへ命令に関する情報を受け渡すパイプラインレジスタ221～223と、デコードされた命令がメモリアクセス命令である場合にそのデコード情報を選択するマルチプレクサ230と、デコードされた命令が分岐命令である場合にそのデコ

ード情報を選択する 2 3 1 と、複数の汎用レジスタからなるレジスタファイル 2 5 1 と、プログラム及びデータを記憶するメモリ 2 5 2 と、フォワーディングユニット 2 5 3 とから構成される。

【 0 0 3 0 】

ここで、命令フェッチ制御部 2 0 1 は、命令デコーダ 2 0 2 ～ 2 0 4 に対して命令アドレスの若い順に命令を配置するよう命令を発行する。これは、命令キャンセル部 2 2 0 と相俟って、同時実行される複数命令について、命令デコーダ 2 0 2 ～ 2 0 4 の順に命令を実行した場合と同じ結果を得るように構成されているからである。

【 0 0 3 1 】

図 5 は、V L I W プロセッサ 2 0 0 のパイプライン処理の流れを示す図である。同図において、縦軸（下向き）は実行すべきプログラム中の配置された命令の配置順を、横軸はサイクル数を示す。図中、D C 1 ～ D C 3 は命令デコーダ 2 0 2 ～ 2 0 4 に、E X 1 ～ E X 3 は A L U 2 0 5 ～ 2 0 7 に、M E M はメモリアクセス制御部 2 0 9 において処理されることを示す。w a i t は、インターロックの発生を意味する。T 1、T 2・・・はステージ時間を単位とするサイクルである。また、命令デコーダ 2 0 2 及び A L U 2 0 5 を経て命令が処理される経路（パイプラインステージでは I F 1 → D C 1 → E X 1 → W B 1 となる）をスロット x と呼ぶ。同様に I F 2 → D C 2 → E X 2 → W B 2 の経路をスロット y、I F 3 → D C 3 → E X 3 → W B 3 の経路をスロット z と呼ぶ。スロット x 中の命令を命令 x と呼ぶ。スロット y、z についても同様に命令 y、z と呼ぶ。

【 0 0 3 2 】

図 6 は、図 5 に示した命令 1 ～ 5 の一例を示す。同図では各命令について、命令のアドレス、ニーモニック表記、命令の実行内容、命令実行により更新されるリソースとを記載してある。ここでリソースはレジスタやメモリを含む。同図では関連するリソースのみを示してある。ニーモニック表記の s u b、a d d、l d、s t、o r は、それぞれ減算命令、加算命令、メモリからレジスタにデータを書き込むロード命令、レジスタデータをメモリに格納するストア命令、論理和命令を意味する。R 0 ～ R 6 はレジスタ 0 ～ 6 を、(R 4 +) の + (プラス) は

実行後に4を加算するポストインクリメントを意味する。また、命令の実行内容の表記において、 $=$ は代入を、 $mem(R4)$ はR4の内容をアドレスとするメモリのデータを、 $|$ は論理和をそれぞれ意味する。

【0033】

また、ストア命令のMEMステージは2サイクルかかり、ロード命令のMEMステージ及び他の命令のEXステージは1サイクルで終わるものとする。以下では、ストア命令のようにパイプライン段数(VLIWプロセッサ200では4つ)よりも多くのサイクルを要する命令を遅延命令と呼び、パイプライン段数と同数のサイクルで処理される命令を通常命令と呼ぶ。

【0034】

なお、説明を簡単にするためストア命令のMEMステージは2サイクル、ロード命令のMEMステージは1サイクルで終わるものとするが、ロード命令及びストア命令のMEMステージのサイクル数が動的に変化する場合でも、本発明においてなんら問題ない。すなわち、ターゲットプロセッサでは、MEMステージにおいてアクセス先(メモリ素子やi/o)からの応答(ACK)をサイクル毎に待つ、応答があったサイクルでメモリアクセスを終えるようにしている。

【0035】

命令群1は、通常命令である命令1~3からなり、図5に示すようにT1~T4の4サイクルで処理される。命令群2は、通常命令である命令5と遅延命令である命令4を含むので、インターロックの発生により5サイクルの処理時間を要する。

【0036】

図7は、図6の命令列についてのサイクルと更新されるリソースと関係を示す図である。サイクルN+1では、命令群1の命令1~3が4サイクルで完了するので、命令1~3によりデスティネーションとして指定されたレジスタR0、R2~R4が更新される。サイクルN+2では、インターロックにより命令群2がまだ完了していないので、更新されるリソースが存在しない。サイクルN+3では、命令群2が完了するので、命令4、5によりデスティネーションとして指定された $mem(R4)$ 、R4、R5が更新される。

【0037】

図8は、命令キャンセル部220により命令がキャンセルされる場合の命令列の例を示す図である。同図において命令6、7は同時実行される命令であり、命令8は単独実行される命令である。命令6の比較命令（cmp命令）は、フラグF0を比較結果に応じてリセット（無効化）する。命令7の加算命令（add命令）は、フラグF0が1であれば実行されるが、フラグF0が0であれば実行されない。

【0038】

命令キャンセル部220は、同時実行している命令6、7について、命令6の結果に応じて、命令7をキャンセルする。つまり、命令キャンセル部220は、命令6によってフラグF0が0にリセットされた場合には、命令7の実行結果をレジスタ又はメモリへ書き込みを禁止することにより、その命令7をキャンセルする。

【0039】

言い換えれば、命令キャンセル部220は、同時実行している複数命令中に、条件分岐命令がある場合に条件の成否に応じて条件分岐命令に後続する命令をキャンセルすることできる。こうして、キャンセル部220は、同時実行される複数命令中の任意の命令から条件分岐することを可能にしている。例えば、図3に示した停止行マークのある条件文（if文）は、停止命令マークのあるcmp命令とフラグF0付きのmov命令として実行される（この2命令は命令6、7と同じ使い方である）。

【0040】

上記の命令6、7は同時に実行される命令であることから本来は先後関係を有していないが、命令キャンセル部220は、論理的に命令6の実行が先行しているものとして扱っている。

【0041】

また、命令キャンセル部220は、複数命令中に条件のない単純分岐命令が存在し、分岐により実行されることのない命令が同時に発行された場合でも、その命令をキャンセルする。

【0042】

命令キャンセル部220を備えることによりVLIWプロセッサ200は、複数命令を同時実行するプロセッサでありながら、同時実行される複数命令中の任意の命令からの条件分岐可能にするため、同時実行される複数命令の間の関係において命令アドレスの順に命令を実行した場合と同じ結果を得るというアーキテクチャとなっている。

【0043】

図9は、命令キャンセル部220により命令がキャンセルされる場合の命令列について他の例を示す図である。同図において、命令12～14は同時実行される命令である。ところが命令13のロード命令(1d命令)も命令14の転送命令(mov命令)もレジスタR1をデスティネーションとして指定しており、両命令によるレジスタR1への書き込みが競合する関係にある(この関係を出力依存と呼ぶ)。出力依存関係を検出した場合に、命令キャンセル部220は、命令アドレスが若い方の命令13によるレジスタR1への書き込みをキャンセルする。その結果、レジスタR1には命令14の実行結果が反映されることになる。この実行結果は、VLIWプロセッサ200が、同時実行する複数命令の間の関係において命令アドレスの順に命令を実行した場合と同じ結果を得るというアーキテクチャを採用していることによる。

【0044】

図10は、フォワーディングユニット253によるフォワーディングを伴うパイプライン処理を示す説明図である。同図において、ロード命令(1d R2, (R3+))及び加算命令(add R4, 8)は同時に発行され、分岐命令(br R)は単独で発行されるものとする。また、ロード命令はMEMステージが2サイクルを要するものとする。

【0045】

加算命令(add R4, 8)の実行結果R4が、分岐命令(br R4)により使用されるので、この2つの命令はデータ依存関係を有している。もし、加算命令のWBステージの完了を待って分岐命令を開始するとすれば、データ依存関係に起因して2サイクルのペナルティが発生することになる(DC1ステージが

T6のサイクルになる)。

【0046】

これを回避するため、フォワーディングユニット253は、現在実行中の命令群中の何れかの命令（被依存命令と呼ぶ）と次の命令群中の何れか命令（依存命令と呼ぶ）との間にデータ依存関係がある場合に、被依存命令のEXステージにおける実行結果として得られたデータを取り込んで一旦保持し、依存命令のEXステージ又はMEMステージの開始時に当該データをオペランドデータとして直接出力する（これをフォワーディングと呼ぶ）。

【0047】

さらに、WBステージで書かれた内容を同じサイクルのDCステージで読み出せる構成（Read After Write）にすることで、WBステージの完了を待つことなくデータ依存が存在する後続命令をペナルティなしで実行することができる。

【0048】

図10の場合、被依存命令（add R4, 8）のEX2ステージにおける実行結果として得られたR4のデータ“8”は、フォワーディングユニット253によってDC1ステージにおいて依存命令（br R4）のオペランドで指定されたアドレス（8）としてマルチプレクサ231を介して命令フェッチ制御部201に出力される。ここで、DC1ステージにおいてデータが出力されているのは、図4に示したようにデコード情報（DC情報）がDCステージの途中からIFステージに出力される構成だからである。このように、フォワーディングユニット253を備えることによりVLIWプロセッサ200では、データ依存関係に起因するペナルティを解消している。

【0049】

また、図10において、ロード命令（ld R2, (R3+)）のMEMステージが2サイクルであることからサイクルT4においてメモリアクセスが完了する。また加算命令（add R4, 8）はT3において加算が完了する。この点で、両命令は命令アドレスの順とは異なる順で実行されたことになる。もしEX3ステージをサイクルT4で実行したとすれば、当該2命令を命令アドレス順に実行することにはなるが、依存関係がある場合には依存命令（br R4）にペナ

ルティが生じることになり、遅延することとなる。そのため、V L I Wプロセッサ 2 0 0では例外的に命令アドレスの順とは異なる順で実行することがある。その場合でも、実行結果は命令アドレス順に実行した場合と同じ結果を得ることになる。

【 0 0 5 0 】

なお、ターゲットプロセッサは図 4に限るものではなく、複数の命令を同時に実行するものであればよい。例えば、図 1 1に示すV L I Wプロセッサでもよい。図 1 1のV L I Wプロセッサは、図 4に示したV L I Wプロセッサから命令キャンセル部 2 2 0を削除した構成となっている。

【 0 0 5 1 】

<シミュレーションシステム 1 の構成>

図 4～図 1 1に示したターゲットプロセッサを前提に、本発明の実施の形態におけるシミュレーションシステム 1 についての説明を続ける。

【 0 0 5 2 】

図 1 2は、シミュレーションシステム 1 の構成を示す機能ブロック図である。同図においてシミュレーションシステム 1 は、ユーザインターフェース 4、デバッガ 3 a、シミュレーション装置 2 から構成される。

【 0 0 5 3 】

ユーザインターフェース 4 は、図 1 に示した表示装置 3 b 及び入力装置 3 c に相当し、ユーザから命令単位のシミュレーション実行や命令単位のステップ実行を指示する操作を受け付け、そのシミュレーション結果を表示する。もちろん、ユーザインターフェース 4 は、命令単位ではなく同時実行される複数命令を同時実行するサイクル単位のシミュレーション実行の操作も受け付けた場合は、そのシミュレーション結果を表示する。

【 0 0 5 4 】

デバッガ 3 a は、図 1 に示した本体装置 3 a に相当し、ユーザインターフェース 4 を介して命令単位のステップ実行の指示やブレークポイントの指定などの各種制御コマンド 1 2 8 を受け、各種制御コマンドに対する応答としてユーザインターフェース 4 に介して命令単位のシミュレーション結果を反映したレジスタフ

ファイルの内容 1 2 9 やメモリの内容 1 3 0 を表示装置 3 b を介して表示する。

【 0 0 5 5 】

また、デバッガ 3 a は、シミュレーション装置 2 に対して命令単位のシミュレーション実行指示 1 3 1 や、メモリアドレス及びサイズ 1 3 4 をシミュレーション装置 2 に送信し、シミュレーション実行指示 1 3 1 やメモリアドレス及びサイズ 1 3 4 に対する応答としてシミュレーション装置 2 から停止命令通知 1 3 2、レジスタデータ 1 3 3、メモリ内容 1 3 5 を受信する。

【 0 0 5 6 】

シミュレーション装置 2 は、図 1 に示した本体装置 2 a に相当し、図 4 に示したターゲットプロセッサの動作をサイクル単位で複数命令（命令群）の同時実行をシミュレーションするパイプラインシミュレーション部 1 0 と、そのシミュレーション結果に基づいて命令群中の命令単位のシミュレーション結果を生成する命令シミュレーション部 3 0 とを備える。パイプラインシミュレーション部 1 0 による命令群単位のシミュレーション結果から命令シミュレーション部 3 0 において命令毎の実行前後の状態を生成するという二段階のシミュレーションにより、シミュレーション装置 2 は、デバッガ 3 a に対して、あたかも命令単位にシミュレーションしているかのように命令単位のシミュレーション結果を生成する。

【 0 0 5 7 】

パイプラインシミュレーション部 1 0 は、第 1 レジスタファイルモジュール 1 1 と、メモリモジュール 1 2 と、共通情報格納部 1 3 と、フェッチモジュール 1 4 と、フェッチ情報格納部 1 5 と、デコードモジュール 1 6 と、デコード情報格納部 1 7 と、実行モジュール 1 8 と、実行情報格納部 1 9 と、完了処理モジュール 2 0 と、完了情報格納部 2 1 と、過去状態更新制御部 2 2 と、スケジューリングモジュール 2 3 とを備える。

【 0 0 5 8 】

命令シミュレーション部 3 0 は、第 2 レジスタファイルモジュール 3 1、メモリ値退避部 3 2、リソース情報変更部 3 3、シミュレーション制御部 3 4 から構成される。

【 0 0 5 9 】

まず、パイプラインシミュレーション部10及び命令シミュレーション部30の各構成要素を説明するのに先立って、同図における各矢線の意味について説明する。

【0060】

101はシミュレーション制御部34からスケジューリングモジュール23へ出力される1サイクル分の命令群のシミュレーション実行指示、102はそのシミュレーション実行指示に対する応答である。103は命令群のシミュレーション実行前の状態の一部として第1レジスタファイル11から第2レジスタファイルモジュール31へコピーされるレジスタデータである。

【0061】

104はメモリアドレス及びサイズ、105は104にアドレス指定されたメモリデータであって、もし104がストア命令のストア先アドレスであれば当該ストア命令によるストア前のメモリ内容である。106はメモリモジュール12に供給されるメモリアドレス及びサイズ、107は106にアドレス指定されたメモリデータである。108、109は104、105と同内容のデータであり、実行モジュール18からスケジューリングモジュール23に供給される。110は第2レジスタファイルモジュールのレジスタファイルの内容であり、111は第1レジスタファイル11の内容である。

【0062】

112は実行モジュール18からの1サイクルのEXステージのシミュレーションを実行したことを通知する命令実行通知である。113はレジスタ番号及びR/Wの区別、114は113によって指定されたレジスタデータである。115は第2レジスタファイルモジュール31を更新してもよいか否かを問う問い合わせ、116は第2レジスタファイルモジュール31の更新を禁止するか否かを示す更新禁止通知である。117はメモリモジュール12に対するアドレス、サイズ、R/Wの区別であり、118はメモリ内容、117により指定されたメモリデータである。119はフェッチ情報、120はデコード情報、121は実行情報、122は完了情報である。

【0063】

1 2 3 ~ 1 2 6 は、それぞれ完了処理モジュール、実行モジュール、デコードモジュール、フェッチモジュールに対して出力される実行指示であって、この順に出力される（正確には各モジュールが呼び出される）。1 2 7 は、各モジュールで共通に使用される共通情報に含まれるインターロックフラグの値を示す。インターロックフラグはインターロックの発生を意味する。共通情報には、インターロックフラグの他に、パイプラインストールの発生を意味するストールフラグや、分岐命令で指定された分岐先アドレスなどを含む。1 2 8 は各種制御コマンド、1 2 9 は表示用のレジスタデータ、1 3 0 は表示用のメモリデータである。1 3 1 は命令単位のシミュレーション実行指示又はステップ実行指示であり、1 3 2 は 1 3 1 への応答として通知される停止命令通知であり、1 3 3 は当該停止命令の実行前の状態を示すレジスタデータ、1 3 4 はメモリアドレス及びサイズ、1 3 5 は 1 3 4 によってアドレス指定されたメモリ内容であって当該停止命令の実行前の状態を示す。

【 0 0 6 4 】

続いて、パイプラインシミュレーション部 1 0 及び命令シミュレーション部 3 0 の各構成要素について説明する。

【 0 0 6 5 】

<パイプラインシミュレーション部 1 0>

第 1 レジスタファイルモジュール 1 1 は、ターゲットプロセッサのレジスタファイルと同じレジスタ構成を有する。

【 0 0 6 6 】

メモリモジュール 1 2 は、ターゲットプロセッサのメモリ構成を有し、デバッグ対象のプログラムを格納する。

共通情報格納部 1 3 は、インターロックが発生したことを示すインターロックフラグや、特定の命令のデータ依存に起因するパイプラインストールの発生を示すストールフラグ等からなる共通情報を格納する。インターロックフラグは、インターロックを発生させたモジュールによってセット／リセットされ、各モジュールから参照される。具体的には、インターロックフラグは、実行モジュール 1 8 によってストア命令などの遅延命令における 1 サイクル目にセットされ、2 サ

イクル目でリセットされる。各モジュールは、スケジューリングモジュール23から1サイクルの実行指示123～126を受けたとき、インターロックフラグがセットされている場合はwait動作を行う。

【0067】

＜フェッチモジュール＞

フェッチモジュール14は、スケジューリングモジュール23から実行指示126を受けたとき、ターゲットプロセッサのIFステージの1サイクルの動作をシミュレーションする。すなわち、フェッチモジュール14は、メモリモジュール12から、同時実行すべき複数命令（ここでは最大3命令とする）をフェッチし、フェッチ情報119としてフェッチ情報格納部15に格納する。もし、フェッチ情報格納部15の命令が全て有効であればフェッチ情報格納部15にフェッチ情報を格納しない。これは、フェッチ情報格納部15に格納されているデコードされていないフェッチ情報を更新しないためである。

【0068】

フェッチ情報の一例を図13に示す。同図においてフェッチ情報は、命令X～Zと、命令X～Zに対応する有効フラグ及び命令PCとを含む。ここで、「命令X」はターゲットプロセッサでは命令デコーダ202に発行される命令、スロットxに発行される命令xの命令コードである。同様に「命令Y」、「命令Z」は、それぞれ命令デコーダ203、204に発行される命令y、zの命令コードである。それゆえ命令X、Y、Zは命令アドレスの若い順になる。「有効フラグ」は、対応する命令が有効か否かを示す。同時実行すべき命令が3つの場合は命令X～Zの3つの有効フラグが、2つの場合は命令XとYの2つ有効フラグが、単独で命令を実行すべき場合は命令Xの有効フラグが1（有効）になる。「命令PC」は、ターゲットプロセッサにおけるフェッチプログラムカウンタの内容に相当する命令アドレスを意味する。

【0069】

フェッチ情報格納部15は、図13に示したフェッチ情報を格納するためのメモリ領域である。フェッチ情報は、フェッチモジュール14とデコードモジュール16とによって参照及び変更される。

【0070】

<デコードモジュール>

デコードモジュール16は、スケジューリングモジュール23から実行指示125を受けたとき、インターロックフラグ127が1でなければ、ターゲットプロセッサのDCステージの1サイクルの動作をシミュレーションする。すなわち、デコードモジュール16は、フェッチ情報格納部15からフェッチ情報を読み出して解読し、解読結果をデコード情報120としてデコード情報格納部17に格納する。その際、フェッチ情報格納部15内のフェッチ情報に対して解読済命令の有効フラグを0（無効）にする。ただし、有効フラグが0の命令は、フェッチ情報中の当該命令の情報をそのままデコード情報に含める。もし、インターロックフラグが1であれば、デコード情報格納部17のデコード情報を更新しない。

【0071】

デコード情報の一例を図14に示す。同図のようにデコード情報は、フェッチ情報と比べて、命令X～Yそれぞれに対応するレジスタ更新情報と、メモリアクセス命令と、有効フラグと、PCと、レジスタ更新情報と、メモリアクセスアドレスと、メモリアクセスデータと、R/W情報とを新たに追加された点が主として異なっている。以下、フェッチ情報と同じ点は説明を省略して主に異なる点を説明する。

【0072】

「命令PC」は、ターゲットプロセッサにおけるフェッチプログラムカウンタではなくデコードプログラムカウンタの内容に相当する命令アドレスを意味する。「命令Xのレジスタ更新情報」は、命令Xによって更新されるレジスタ（デスティネーションレジスタ）を示す。命令Y、Zのレジスタ更新情報もそれぞれ同様である。この情報は出力依存の検出に用いられる。

【0073】

同図中のメモリアクセス命令以下に列挙された各情報は、命令X～Zの何れかがメモリアクセス命令である場合にのみ有効であり、命令X～Zの何れもがメモリアクセス命令でない場合には無効である。「メモリアクセス命令」は、命令X

～Zの何れかと同じ命令であり、ターゲットプロセッサでは命令デコーダ202～204の何れかからマルチプレクサ230、231を介してメモリアクセス制御部209に発行される命令である。「有効フラグ」は、メモリアクセス命令が有効か否かを示し、デコードモジュール16により初期値として1（有効）にセットされる。「命令PC」は、ターゲットプロセッサにおけるデコードプログラムカウンタの内容に相当する命令アドレスを意味する。「レジスタ更新情報」は当該メモリアクセス命令により更新されるレジスタを示す。「メモリアクセスアドレス」はアクセス先のメモリアドレスを示す。「R/W情報」は、ロード命令の場合はリード、ストア命令の場合はライト、メモリアクセス命令が存在しない場合はNOPを示す。例えば、命令Zがロード命令（ld R0, (R1+)）の場合、命令Zの命令PCとメモリアクセス命令の命令PCは同じ値であるが、デコード情報中の命令Zには（R1=R1+4）を意味する操作コードが、メモリアクセス命令には（R0=mem（R1））を意味する操作コードが設定される。命令Zのレジスタ更新情報はR1、メモリアクセス命令のレジスタ更新情報はR0となる。この場合、命令Zの操作コードとメモリアクセス命令の操作コードとは、演算機能を分担しており、ターゲットプロセッサのALU207とメモリアクセス制御部209の演算機能の分担に対応している。

【0074】

デコード情報格納部17は、図14に示したデコード情報を格納するためのメモリ領域である。デコード情報は、デコードモジュール16によって書き込まれ、実行モジュール18によって読み出される。

【0075】

<実行モジュール18>

実行モジュール18は、スケジューリングモジュール23から実行指示124を受けたとき、ターゲットプロセッサのEX/MEMステージの1サイクルの動作をシミュレーションする。すなわち、実行モジュール18は、デコード情報格納部17からデコード情報120を読み出し、有効フラグが1（有効）の命令について、当該命令の演算内容のシミュレーション（具体的には当該命令に対応する命令実行関数を呼び出し）を行って第1レジスタファイルモジュール11を更

新する。ここで、命令 X、Y、Z はこの順番でシミュレーションし、各命令のシミュレーションが終わる毎に実行モジュール 1 8 は命令 X、Y、Z の何れの命令を実行したかを通知する命令実行通知 1 1 2 を出力する。

【 0 0 7 6 】

デコード情報中に遅延命令（例えば 2 サイクルの MEM ステージを要するメモリアクセス命令）が有効に存在する場合、実行モジュール 1 8 は、遅延命令の遅延が残っていれば（つまり 1 サイクル目）、複数命令中の当該メモリアクセス命令以外の命令 X～Z についてはシミュレーションを行い、メモリアクセス命令についてはシミュレーションを行わないで、共通情報格納部 1 3 中のインターロックフラグを 1 にセットして当該サイクルのシミュレーションを終える。また、遅延命令の遅延が残っていなければ（つまり 2 サイクル目）、当該メモリアクセスのシミュレーションをメモリモジュール 1 2 に対して行い、インターロックフラグをリセットする。その際、メモリアクセス命令がメモリライト命令である場合には、実行情報の一部とするためライト前のデータを読み出しておく。

【 0 0 7 7 】

このようなシミュレーションの結果、実行モジュール 1 8 は実行情報を実行情報格納部 1 9 に格納する。また、デコード情報格納部 1 7 内デコード情報に対してシミュレーション済の命令の有効フラグを 0（無効）にする。

【 0 0 7 8 】

実行情報の一例を図 1 5 に示す。同図の実行情報は、図 1 4 に示したデコード情報と比べて、ストア前メモリ内容が追加されている点が主として異なる。以下、デコード情報と同じ点は説明を省略して主に異なる点を説明する。

【 0 0 7 9 】

命令 X～Y の「命令 PC」及びメモリアクセス命令の「命令 PC」は、ターゲットプロセッサにおける実行プログラムカウンタの内容に相当する命令アドレスを意味する。「ストア前データ」は、メモリアクセス命令がストア命令である場合にメモリライトする前のメモリデータであり、メモリ値退避部 3 2 に出力され、メモリアクセス命令実行前の状態を復元するのに利用される。

【 0 0 8 0 】

実行情報格納部 19 は、図 15 に示した実行情報を格納するためのメモリ領域である。実行情報は、実行モジュール 18 と完了処理モジュール 20 とによって参照及び更新される。

【0081】

＜完了処理モジュール＞

完了処理モジュール 20 は、スケジューリングモジュール 23 から実行指示 123 を受けたとき、インターロックフラグ 127 が 1 でなければ、ターゲットプロセッサの WB ステージの 1 サイクルの動作をシミュレーションする。すなわち、完了処理モジュール 20 は、実行情報格納部 19 から実行情報を読み出し、有効フラグが 1（有効）の命令について、WB（ライトバック）動作を行って、完了情報を完了情報格納部 21 に格納する。また、実行情報格納部 19 内の実行情報に対して完了した命令の有効フラグを 0（無効）にする。

【0082】

ただし、本実施例では、メモリアクセス命令以外の命令におけるレジスタへのライトバックは実行ステージにおいて完了しているので、ほとんどの命令が完了処理を必要としない。

【0083】

完了情報の一例を図 16 に示す。同図において完了情報は、図 15 に示した実行情報と同様なので説明を省略する。ただし、「命令 PC」はターゲットプロセッサにおける完了プログラムカウンタの内容に相当する命令アドレスを意味する。

【0084】

完了情報格納部 21 は、図 16 に示した実行情報を格納するためのメモリ領域である。

過去状態更新制御部 22 は、スケジューリングモジュール 23 からの問い合わせ 115 に対して、共通情報格納部 13 に格納されたインターロックフラグが 1 である場合には、第 1 レジスタファイルモジュール 11 から第 2 レジスタファイルモジュール 31 へのレジスタファイルのデータのコピーを禁止する旨の更新禁止通知 116 を応答し、インターロックフラグが 0 である場合には、コピーを禁

止しない旨の更新禁止通知 1 1 6 を応答する。ここで、第 2 レジスタファイルモジュール 3 1 は、通常 1 サイクル前の第 1 レジスタファイルモジュール 1 1 の内容が保存されている。過去状態更新制御部 2 2 が禁止する旨を通知するのは、インターロックの発生により EX/MEM ステージが 2 サイクルかかる場合に、第 2 レジスタファイルモジュール 3 1 には、EX/MEM ステージの開始前の状態のレジスタデータを保存しておくためである。

【 0 0 8 5 】

<スケジューリングモジュール>

スケジューリングモジュール 2 3 は、シミュレーション実行指示 1 0 1 を受けたとき、複数命令を同時実行する 1 サイクル分のパイプライン処理をシミュレーションするようスケジューリングし、1 サイクルのシミュレーション完了後に応答 1 0 2 を命令シミュレーション部 3 0 に出力する。

【 0 0 8 6 】

図 1 7 は、スケジューリングモジュール 2 3 のスケジューリングによる命令群のシミュレーション処理を示すフローチャートである。

同図に示すように、スケジューリングモジュール 2 3 は、シミュレーション制御部 3 4 からシミュレーション実行指示 1 0 1 を受けたとき (S 1 1 : yes)、過去状態更新制御部 2 2 を呼び出すことにより問い合わせ 1 1 5 をかけ (S 1 2)、その応答として更新禁止通知 1 1 6 が第 2 レジスタファイルモジュール 3 1 の更新を禁止していなければ (S 1 3 : yes) 第 1 レジスタファイルモジュール 1 1 の内容を第 2 レジスタファイルモジュール 3 1 にコピーし (S 1 4)、禁止していれば (S 1 3 : no) コピーしないで、実行指示 1 2 3、1 2 4、1 2 5、1 2 6 をこの順で出力する (S 1 5 ~ 1 8)。実行指示 1 2 3 ~ 1 2 6 はシミュレーションプログラム中では関数呼び出しという形式で実現される。したがって、完了処理モジュール 2 0、実行モジュール 1 8、デコードモジュール 1 6、フェッチモジュール 1 4 の順に実行されることになり、これにより命令群の 1 サイクル分のパイプライン処理がなされることになる。さらに、スケジューリングモジュール 2 3 は、1 サイクルのパイプライン処理が完了したことを通知する応答 1 0 2 をシミュレーション制御部 3 4 に出力する。

【0087】

<命令シミュレーション部30>

第2レジスタファイルモジュール31は、第1レジスタファイルモジュール11の1サイクル以上前（つまり一命令群の実行前）のレジスタデータのコピーを保持する。この保持内容は、一命令群の実行後に当該命令群に含まれる各命令についての実行前のレジスタデータを復元するために用いられる。

【0088】

メモリ値退避部32は、実行モジュール18においてストア命令のシミュレーションが実行される場合に、当該メモリ命令の書き込み先アドレスのストア前のメモリ値を退避させて保持する。

【0089】

リソース情報変更部33は、シミュレーション制御部34から停止命令の通知を受けたとき、当該停止命令のシミュレーション実行前のリソースの状態を復元する。この停止命令は、パイプラインシミュレーション部10によるシミュレーション実行済の最後の命令群に含まれる何れか1つの命令が指定される。リソース情報変更部33は、パイプラインシミュレーション部10による命令群のシミュレーション実行後のリソースと実行前のリソースとに基づいて、通知された停止命令がまだシミュレーションされていなかったとしたときのリソース（メモリデータ及びレジスタデータ）状態を復元する。言い換えれば、停止命令の直前の命令までシミュレーションがなされた場合に対応するリソース状態を復元する。ここで、命令群のシミュレーション実行後のリソースは第1レジスタファイルモジュール11及びメモリモジュール12に保持されている。また、当該命令群のシミュレーション実行前のリソースは第2レジスタファイルモジュール31、メモリ値退避部32に保持されている。

【0090】

具体的には、停止命令が命令Xであるとき、リソース情報変更部33は、命令群中の命令命令X、Y及びZのシミュレーション前の状態を復元する。停止命令が命令Yであるとき、リソース情報変更部33は、命令群中の命令Y及びZのシミュレーション前の状態を復元し、停止命令が命令Zであるときは命令群中の命

令 Z のシミュレーション前の状態を復元する。復元に際して、リソース情報変更部 3 3 は、図 1 4 に示した実行情報 1 2 1 を参照して、命令 X、Y、Z、メモリアクセス命令のうち有効な命令によって更新されるリソースを判別し、命令 X、Y、Z、メモリアクセス命令それぞれのシミュレーション前のリソースを取得する。取得したデータのうち停止命令の直前の状態を示すデータをレジスタデータ 1 3 3 やメモリ内容 1 3 5 としてデバッガ 3 a に出力する。

【 0 0 9 1 】

シミュレーション制御部 3 4 は、停止命令がどれであるか示す停止命令ポインタを保持し、デバッガ 3 a からシミュレーション実行指示 1 3 1 に従って、命令単位のシミュレーション実行を制御し、デバッガ 3 a にその結果として停止命令通知を送信する。すなわち、シミュレーション制御部 3 4 は、パイプラインシミュレーション部 1 0 による命令群のシミュレーションにおいて直前に実行済の複数命令の何れか 1 つを停止命令として管理しており、シミュレーション実行指示 1 3 1 を受けたとき、ブレーク条件に合致する命令がパイプラインシミュレーション部 1 0 による直前にシミュレーション済の命令群に存在する場合には、停止命令ポインタをブレーク条件に合致する命令に更新し、停止命令通知 1 3 2 をデバッガ 3 a 及びリソース情報変更部 3 3 に出力する。一方、ブレーク条件に合致する命令がパイプラインシミュレーション部 1 0 による直前にシミュレーション済の命令群に存在しない場合は、パイプラインシミュレーション部 1 0 に命令群のシミュレーションを 1 サイクル進めるようにシミュレーション実行指示 1 0 1 を出力する。こうして、ブレーク条件に合致する命令が直前の命令群のシミュレーション結果に存在するようになるまで、シミュレーション実行指示 1 0 1 を出力する。

【 0 0 9 2 】

図 1 8 は、シミュレーション制御部 3 4 による命令単位のシミュレーション制御を示すフローチャートを示す。

同図では、シミュレーション制御部 3 4 は、デバッガ 3 a からシミュレーション実行指示 1 3 1 を受けると (S 2 1)、当該指示 1 3 1 がステップ実行指示であれば (ブレーク条件の指定がなければ)、ブレーク条件を次の「実行済みの命

令」と設定する（S 2 3）。次に、シミュレーション制御部 3 4 は、現在の停止命令が命令 Z に該当する場合には（S 2 4 : y e s）、スケジューリングモジュール 2 3 にシミュレーション実行指示 1 0 1 を出力し、応答 1 0 2 の受信するまで待ち（S 3 0）、応答受信後にサイクル数を + 1 インクリメントする。これにより、パイプラインシミュレーション部 1 0 において新たに次の命令群のシミュレーションがなされる。また、このサイクル数は、命令単位のシミュレーションのサイクル数でなく、命令群単位のシミュレーションにおけるサイクル数である。これにより、シミュレーション制御部 3 4 は、ターゲットプロセッサにおけるサイクル数を正確にカウントしている。

【 0 0 9 3 】

さらに、シミュレーション制御部 3 4 は、シミュレーションされた新たな命令群における命令 X が実行済で且つブレイク条件に合致するか否かを判定する（S 3 2）。命令 X が実行済であるか否かの判定については、シミュレーション制御部 3 4 が実行情報格納部 1 9 に格納された実行情報中の命令 X の有効フラグが 1（有効）でありかつ共通情報格納部 1 3 に格納されたインターロックフラグが 0（インターロック中でない）であれば実行済と判定する。有効フラグが 0（無効）であれば、当該命令群中に有効な命令 X が存在しないし、インターロックフラグが 1（インターロック中）であれば、命令 X が存在していても命令群のシミュレーションが完了していないからである。命令 Y、Z が実行済か否かの判定についても同様である。

【 0 0 9 4 】

S 3 2 の判定の結果、命令 X が実行済で且つブレイク条件に合致する場合には、停止命令ポインタを命令 X に更新し（S 3 3）、シミュレーション制御部 3 4 は、リソース情報変更部 3 3 及びデバッガ 3 a に更新された停止命令を停止命令通知 1 3 2 として通知する（S 3 4）。S 3 2 の判定の結果、命令 X が実行済でないか、またはブレイク条件に合致しない場合には、S 2 6 に進む。

【 0 0 9 5 】

また、シミュレーション制御部 3 4 は、現在の停止命令が命令 Y であり（S 2 5 : y e s）、命令 Z が実行済で且つブレイク条件に合致するか否かを判定する

(S28)。S28の判定の結果、命令Zが実行済で且つブレーク条件に合致する場合には、停止命令ポインタを命令Zに更新し(S29)、リソース情報変更部33及びデバッガ3aに更新された停止命令を停止命令通知132として通知する(S34)。S28の判定の結果、命令Zが実行済でない、またはブレーク条件に合致しない場合には、S30に進む。

【0096】

また、シミュレーション制御部34は、現在の停止命令が命令Xであるとき(S25で命令Yでないと判定されたとき)、命令Yが実行済で且つブレーク条件に合致するか否かを判定する(S26)。S26の判定の結果、命令Yが実行済で且つブレーク条件に合致する場合には、停止命令ポインタを命令Yに更新し(S27)、リソース情報変更部33及びデバッガ3aに更新された停止命令を停止命令通知132として通知する(S34)。S26の判定の結果、命令Yが実行済でない、またはブレーク条件に合致しない場合には、S38に進む。

【0097】

このように、シミュレーション制御部34は、ブレーク条件に合致する命令が命令群のシミュレーション結果中に見つかるまで、順次パイプラインシミュレーション部10に命令群の1サイクルのシミュレーション実行指示101を出力する。したがって、停止命令が見つかった時点では、その停止命令の属する命令群のシミュレーション後とそのシミュレーション前の状態(リソース)が必ず保存されていることになる。これにより命令単位にシミュレーション実行前後の状態の復元を可能にしている

【0098】

図19は、リソース情報変更部33の構成を示すブロック図である。同図のようにリソース情報変更部33は通常命令結果生成部35、第1補完部36、第2補完部37、メモリ内容選択部38を備え、シミュレーション制御部34から通知される停止命令のシミュレーション実行前の状態を復元する。

【0099】

通常命令結果生成部35は、シミュレーション制御部34から停止命令通知132を受けたとき、第1レジスタファイルモジュール11、メモリモジュール1

2、第2レジスタファイルモジュール31、メモリ値退避部32の内容に基づいて当該停止命令の実行前のレジスタファイルを復元する。通常命令結果生成部35は、通知された命令が通常命令であっても遅延命令であってもレジスタファイルを復元するが、遅延命令である場合や出力依存がある場合には、第1補完部36、第2補完部37によって復元を補完される。

【0100】

通常命令結果生成部35の詳細な構成を示すブロック図を図20に示す。同図のように通常命令結果生成部35は、命令毎レジスタファイル退避部39と、命令Xのシミュレーション実行結果を保持するための第3レジスタファイルモジュール40と、命令Yのシミュレーション実行結果を保持するための第4レジスタファイルモジュール41を備える。

【0101】

命令毎レジスタファイル退避部39は、実行モジュール18から出力されためいれいXの命令実行通知を受け、第1レジスタファイルモジュール11の内容を、第3レジスタファイルモジュール40に命令Xまで実行した場合のシミュレーション実行結果としてコピーし、同様に命令Yの命令実行通知を受け、第1レジスタファイル11の内容を第4レジスタファイルモジュール41に命令Yまで実行した場合のシミュレーション実行結果としてコピーする。これにより命令X、Y、Zを順次実行した場合の各レジスタデータが第3、第4、第1レジスタファイルモジュールに格納されることになる。このとき、第2レジスタファイルモジュール31は、直前の命令群の実行結果を格納している。また、命令毎レジスタファイル退避部39は、シミュレーション制御部34から停止命令の通知を受けたとき、停止命令が命令Xであればその直前の命令群の実行結果を示す第2レジスタファイルモジュール31の内容を出力する。同様に、停止命令が命令Yであれば第3レジスタファイルモジュール40の内容を、停止命令が命令Zであれば第4レジスタファイルモジュール41の内容を第1補完部36、第2補完部37を介してデバッガ3aに出力する。この出力は、命令X～Zが通常命令（遅延を生じさせない命令）である場合は、第1補完部36、第2補完部37による補完を受ける必要がなく、そのままデバッガ3aに出力される。

【0102】

このように、通常命令結果生成部35は、命令群に遅延命令が含まれない場合には、停止命令が命令X、Y、Zのどれであっても当該停止命令直前の命令が実行済の時点におけるレジスタファイルの内容を生成する。それゆえ、ターゲットプロセッサが同一命令郡中の出力依存を許さないアーキテクチャの場合、第3、第4レジスタファイルモジュール40、41は省略可能である。

【0103】

第1補完部36は、EXステージで遅延命令（メモリアクセス命令）が命令群に含まれている場合に、実行情報格納部19から、当該遅延命令が更新するレジスタ番号と更新内容（メモリアクセスデータ）を参照し、通常命令結果生成部35によって生成された停止命令実行前の状態を示すレジスタファイルの内容を補完する。

【0104】

例えば、ロード命令（ld R0, (R1+)）が2サイクルかかるものとする、1サイクル目ではレジスタR1の更新が完了し、2サイクル目ではレジスタR0の更新が完了しなければならない。なぜならターゲットプロセッサにおいてR1の更新は1サイクルで実行する3つのALUの何れかが分担し、R0の更新は2サイクルを要するメモリアクセス制御部209が分担するからである。そのため、実行情報格納部19は、メモリアクセス命令の更新レジスタ番号と、更新内容であるメモリアクセスデータを保持する。第1補完部36は、停止命令が同一命令群のロード命令よりも後の命令であった場合、実行情報格納部19からメモリアクセス命令のレジスタ更新情報と更新される内容であるメモリアクセスデータとを読み出し、更新されるレジスタを遅延レジスタとして、メモリアクセスデータを遅延データとして認識する。

【0105】

さらに、第1補完部36は、通常命令結果生成部35から出力されたレジスタファイルの内容の遅延レジスタに該当する箇所を遅延データで更新することにより遅延命令が存在する場合のレジスタファイルを復元する。ただし、ロード命令で更新されるレジスタが、同一命令郡中のより後ろの命令により更新される場合

(出力依存がある場合)はロード命令によるレジスタ書き込みがキャンセルされるので補完を行わない。また、遅延命令がレジスタを更新しない命令(ストア命令など)の場合も当然補完を行わない。これにより命令単位のシミュレーションを指示された場合に命令単位でリソースを正確に復元する。

【0106】

第2補完部37は、MEMステージで2サイクル以上を要する遅延命令と、遅延より後に配置された遅延命令と出力依存関係にあるその他の命令(以降出力依存命令と呼ぶ)が同一命令群に含まれる場合において、停止命令より前に遅延命令があり、停止命令以降に出力依存命令がある場合に、第1補完部36と同様に遅延レジスタを遅延データで更新し、第1補完部36から出力されるレジスタファイルの内容を補完する。

【0107】

例えば、命令Yがロード命令(`ld R1, (R2+)`)、命令Zが転送命令(`mov R1, 3`)であって、ロード命令のEXステージが2サイクルを要するものとする。この場合、命令Yと命令ZはともにレジスタR1をデスティネーションとする出力依存関係を有するが、命令ZによるレジスタR1の更新がなされなければならない。なぜならターゲットプロセッサでは、EXステージの1サイクル目においてロード命令によるレジスタR2のインクリメントと`mov`命令によるレジスタR1の更新がなされ、2サイクル目においてロード命令によるレジスタR1の更新が命令キャンセル部220によってキャンセルされるからである。ところが、命令単位のシミュレーションにおいては、まだ実行もされていない命令Zによって命令Yの実行結果がキャンセルされるのは、不自然である。むしろ、命令ZによりレジスタR1が上書きされたことによりキャンセルされたものとするべきであろう。そのため、第2補完部37は、命令Y、Zを順番に1つつ実行した場合と同じ結果を生成するため、出力依存によりキャンセルされる実行内容であっても、その実行結果を補完する。

【0108】

図21は、メモリ値退避部32の詳細な構成を示すブロック図である。同図のようにメモリ値退避部32は、ストア命令によって書き込まれる前のメモリデー

タを保持するためのストア前データ格納部 4 2 と、当該ストア命令によって指定されたアドレスを保持するストアアドレス格納部 4 3 と、メモリ内容変更部 4 4 とを備える。

【 0 1 0 9 】

メモリ内容変更部 4 4 は、リソース情報変更部 3 3 から停止命令通知を受けたとき、命令群中の停止命令以降にストア命令が含まれている場合に、当該ストア命令実行前のメモリ内容を復元する。シミュレーション制御部 3 4 から読み出しを指定されたメモリアドレス及びサイズ 1 0 4 に、当該ストア命令のストア先のアドレスが含まれている場合には、メモリモジュール 1 2 のデータの代わりにストア前データ格納部 4 2 のデータをメモリ内容 1 0 5 に含ませてリソース情報変更部 3 3 に出力する。

以上のように構成された実施の形態におけるシミュレーションシステム 1 について、プログラム例を挙げて説明する。

【 0 1 1 0 】

＜第 1 プログラム例＞

図 2 2 は、シミュレーション対象の第 1 のプログラム例を示す図である。同図のプログラム例は、同時実行される命令 6、7 からなる命令群 1 と命令 8 からなる命令群 2 のみを示している。各命令について命令アドレスを示す「PC」と「ニーモニック」と「シミュレーション結果」と「表示結果」と「停止」とを記している。命令群 1 のシミュレーション直前の状態は、{R 0、R 1、R 2、R 3、F 0} = {1、0、0、0、1} であるものとする。

【 0 1 1 1 】

「シミュレーション結果」は、命令シミュレーション部 3 0 による命令単位で当該命令をシミュレーションした後の状態（同図では R 0 ～ R 3、F 0 のみを記した）を示す。「表示結果」は、当該命令が停止命令であるときにデバッガ 3 に表示される状態であって、当該停止命令の実行前の状態を示す。「停止」は、当該命令をブレイク条件とした場合にブレイク（停止）するか否か、つまり当該命令が停止命令となるか否かを示す。

【 0 1 1 2 】

このプログラム例の場合、命令6、8は停止命令となり得るが、命令7は停止命令にはならない。なぜなら、命令7はフラグF0により無効化されるので、命令7をブレーク条件とするシミュレーションを行っても、ブレークしないことになる。その際、シミュレーションシステム1は、停止命令の検出を、命令のシミュレーション実行前にブレーク条件を判定するのではなく、命令群のシミュレーション実行後にブレーク条件を判定しているからである。これによりターゲットプロセッサにおけるプログラムの実行経路（又は分岐経路）と同じ実行経路（又は分岐経路）をシミュレーションした結果を得ている。

【0113】

より詳しく言うと、パイプラインシミュレーション部10での命令群1のシミュレーションにおいて、実行モジュール18は、命令6のシミュレーションによりフラグF0がリセットされるので、命令7のシミュレーションをキャンセルし、命令7の有効フラグをリセットした実行情報122を実行情報格納部19に格納する。シミュレーション制御部34は、命令群1のシミュレーション結果のうち有効フラグが0ならば命令がまだ実行されていない（または命令が存在しないと）と判断するので、命令7についてブレーク条件の合致するかどうか判定を行わない。それゆえ、命令7は停止命令とはならない。

【0114】

このようにシミュレーションシステム1は、命令群単位にシミュレーションを停止するのではなく、命令単位に停止することができる。しかも、ターゲットプロセッサがキャンセル機構を備える場合には、命令群中の命令キャンセルも正確にシミュレーションする。

【0115】

＜第2プログラム例＞

図23は、シミュレーション対象の第2のプログラム例を示す図である。同図のプログラム例では、同時実行される命令1～3からなる命令群1と命令4、5からなる命令群2を示している。各命令についての「PC」「ニーモニック」「表示結果」「停止」は図22と同様である。ただし、命令群1のシミュレーション直前の状態は、 $\{R0, R1, R2, R3, R4, R5, R6\} = \{10, 5$

、0、0、0、1、2}、mem(0)=100、mem(4)=200であるものとする。

【0116】

このプログラム例中の命令3は、レジスタR4の内容をアドレスとしてメモリデータを読み出してレジスタR3にロードし、さらにレジスタR4を+4インクリメントすることを指示する命令である。命令4は、レジスタR2のデータを、レジスタR4の内容をアドレスとしてメモリにストアし、さらにレジスタR4を+4インクリメントすることを指示する命令である。この命令3と命令4とはデータ依存関係がある。つまり、命令3がレジスタR4を+4インクリメントした結果を命令4は使用しているので、命令3の実行完了後でなければ命令4は正しく実行できない。

【0117】

この点、ターゲットプロセッサでは、図10で説明したフォワーディングにより、インターロックの発生を防止している。すなわち、命令3の実行ステージにおいて+4インクリメントしたレジスタR4のデータを、フォワーディングユニット253により次のサイクルで命令4の実行ステージに供給している。図10ではロード命令の実行ステージが2サイクルかかる場合を前提にしているので、1サイクルのインターロックが発生しているけれども、このインターロックはデータ依存関係に起因するのではなく、2サイクルの実行ステージに起因する。ロード命令の実行ステージが1サイクルだけ場合には、図10においてもインターロックは発生しない。

【0118】

これに対応するシミュレーションシステム1のシミュレーションについて、(1)命令3のロード命令が1サイクルで終了する場合、(2)命令3のロード命令が2サイクルを要する場合に分けて説明する。

【0119】

(1) 命令3(ロード命令)の実行ステージが1サイクルで終了する場合

実行モジュール18は、命令3の実行ステージのシミュレーションにおいて第1レジスタファイルモジュール11のレジスタR4のデータを格納する。さらに

、実行モジュール18は、次のサイクルで命令4の実行ステージのシミュレーションとして第1レジスタファイルモジュール11のレジスタR4を用いて命令4のシミュレーションを行う。この場合、シミュレーションシステム1は、ターゲットプロセッサのフォワーディングとに相当する機能を第1レジスタファイルモジュール11を用いて実現している。

【0120】

一方、ブレーク条件又はステップ実行により命令4を停止命令としている場合に、シミュレーション制御部34は、図18に示したフローの通り、パイプラインシミュレーション部10における命令群2のシミュレーション結果において命令Xが命令4に合致するので、停止命令を命令Xに更新してデバッガ3a及びリソース情報変更部33にその旨通知する。リソース情報変更部33は、命令Xの実行前の状態を復元してデバッガ3aにレジスタデータ133及びメモリ内容135を通知する。このようにして命令X（命令4）の直前の命令3のシミュレーション結果は、図23における命令4の表示結果欄のようになる。

【0121】

また、この状態でmem(4)をデバッガ3aが参照した場合、命令4（ストア命令）の実行前のmem(4)を表示するためリソース情報変更部33はメモリ値退避部32に退避されている値200を表示する。

【0122】

（2）命令3（ロード命令）の実行ステージが2サイクルかかる場合

この場合も、（1）と同様に、図24に示した表示結果を得るが、シミュレーションシステム1において2サイクルのシミュレーションを行う点が（1）と異なる。これは、ターゲットプロセッサのサイクル数についても正確にシミュレーションするためである。

【0123】

詳しくは、命令3はデコード情報格納部17中のデコード情報120においてレジスタR4に対する+4のインクリメントを指示する「命令Z」及びR4をアドレスとしてメモリデータを読み出しR3に格納する「メモリアクセス命令」として設定される。実行モジュール18は、命令3の実行ステージの1サイクル目

のシミュレーションにおいて、当該「命令Z」（レジスタR4の+4インクリメント）のシミュレーションとして第1レジスタファイルモジュール11のレジスタR4を更新すると共にインターロックフラグをセットする。2サイクル目で当該「メモリアクセス命令」をシミュレーションする。

【0124】

さらに、命令3の1サイクル目の次のサイクルでは、命令Zのシミュレーション結果（第1レジスタファイルモジュール11の更新されたレジスタR4）が命令4により利用可能になる。これによりフォワーディングと同様の機能を果たしている。

【0125】

また、一方、ブレーク条件又はステップ実行により命令4を停止命令としている場合に、シミュレーション制御部34は、図18に示したフローに従って、（1）と同様に、命令Xの実行前の状態を復元してデバッガ3aにレジスタデータ133及びメモリ内容135を通知する。

【0126】

上記の（1）（2）はシミュレーション制御部34によって命令単位に停止命令を決定し、リソース情報変更部33によって決定された停止命令が未実行で直前の命令のシミュレーション済の状態を復元するので、同じ結果となる。また、パイプラインシミュレーション部10ではサイクル単位に命令群のシミュレーションを行っているので、シミュレーション制御部34にカウントされるサイクル数は、ターゲットプロセッサにおけるサイクル数を（1）の場合も（2）の場合もそれぞれ正確に正確にカウントすることができる。

【0127】

また、図23における命令4（ストア命令）の実行ステージが1サイクルである場合は（1）と、2サイクルである場合は（2）と同様にして、シミュレーション制御部34はサイクル数を正確にカウントすることができる。

【0128】

＜第3プログラム例＞

図24は、遅延命令が出力依存関係を有している場合のシミュレーション対象

の第3のプログラム例を示す図である。同図のプログラム例では、命令群5に属する命令12～14に加えて、パイプラインシミュレーション部10による命令群5のシミュレーション後の第1レジスタファイルモジュール11、第2レジスタファイルモジュール31、第3レジスタファイルモジュール40、第4レジスタファイルモジュール41の各内容（同図ではR0～R2のみ）と、実行情報格納部19内のメモリアクセスデータの内容を記してある。

【0129】

このプログラム例では、命令13が遅延命令であり、かつ命令13と命令14とが出力依存関係を有している。命令群5のシミュレーション直前の状態は{R0、R1、R2} = {0、0、0} mem(0) = 200であるものとする。また、命令13のメモリアクセスは2サイクルを要するものとする。

【0130】

パイプラインシミュレーション部10及びシミュレーション制御部34の動作は、第2プログラム例の(2)と同様に、遅延命令による2サイクルの実行ステージをシミュレーションする。リソース情報変更部33は、出力依存関係のある命令13、14については、あたかも順に実行したような結果を生成する。つまり、リソース情報変更部33は、同図の命令14の「表示結果」欄のように、直前の命令13のシミュレーション結果を生成する。

【0131】

この点、ターゲットプロセッサでは、命令13によるレジスタR1の更新はキャンセルされ、命令14によるレジスタR1の更新のみ実行される。これは命令12、13の順に実行した結果を得るためである。

【0132】

これに対して、リソース情報変更部33は、命令13の次の命令（命令14）が停止命令であれば、命令13のシミュレーション結果を生成し、命令14の次の命令が停止命令であれば、命令14のシミュレーション結果を生成する。リソース情報変更部33が命令12、13の順に実行した結果を得るという点では、ターゲットプロセッサと同じであるが、リソース情報変更部33はキャンセルされることになる命令13についてもシミュレーション結果を生成する点で異なる

。デバッガ 3 a のユーザにとってはデータ依存関係のある命令が結果的にキャンセルされる過程がわかる点で、使い勝手がよくなっている。

【 0 1 3 3 】

具体的には、パイプラインシミュレーション部 1 0 は、命令群 5 のシミュレーションによって、第 1、第 2 レジスタファイルモジュール 1 1、3 1 を更新する。またこのとき、レジスタファイル退避部 3 9 は実行モジュール 1 8 から命令実行通知を受けて、第 3、4 レジスタファイルモジュール 4 0、4 1 を更新する。

【 0 1 3 4 】

その結果、第 1 レジスタファイルモジュール 1 1 は命令群 5 のシミュレーション直後のデータを保持する。これは、命令 Z（つまり命令 1 4）のシミュレーション直後のデータである。第 2 レジスタファイルモジュール 3 1 は命令群 5 のシミュレーション直前のデータを保持する。第 3 レジスタファイルモジュール 4 0 は命令 X（命令 1 2）のシミュレーション後のデータを、第 4 レジスタファイルモジュール 4 1 は命令 Y（つまり命令 1 3）シミュレーション後のデータを保持する。実行情報格納部 1 9 内のメモリアクセスデータは、命令 1 3 でロードされたメモリの内容を保持する。

【 0 1 3 5 】

命令 1 4 が停止命令として指示されたとき、リソース情報変更部 3 3 は、第 4 レジスタファイルモジュール 4 1 のデータに対して第 1 補完部 3 6 でメモリアクセスデータを補完したデータをレジスタデータ 1 3 3 としてデバッガ 3 a に出力する。

【 0 1 3 6 】

このように、シミュレーションシステム 1 によれば、遅延命令と他の命令とに出力依存関係がある場合、それらの命令を順に実行した命令単位のシミュレーション結果を得ることができ、しかも、命令群単位のサイクル数を正確にカウントすることができる。

【 0 1 3 7 】

以上説明してきたように本発明の実施の形態におけるシミュレーションシステム 1 によれば、複数命令を同時実行するプロセッサをターゲットとするが、命令

単位にシミュレーションを行うことができる。それゆえ、命令群単位のブレークではなく、同時実行される命令単位にブレークすることができる。

【 0 1 3 8 】

しかも、シミュレーションシステム 1 は、命令群のサイクル単位のシミュレーションと命令単位のシミュレーションとからなる 2 段階のシミュレーションを行うので、ターゲットプロセッサのサイクル数を正確にシミュレーションすることができる。

【 0 1 3 9 】

また、ターゲットプロセッサがフォワーディング機構を有している場合、遅延命令によりインターロックが発生する場合、およびキャンセル機構を有している場合でもターゲットプロセッサのサイクル数を正確にシミュレーションすることができる。

【 0 1 4 0 】

なお、上記実施の形態において、シミュレーション制御部 3 4 は、命令単位に命令のシミュレーション後にブレーク条件に合致するか否かを判定しているが、この代わりに、シミュレーション前にブレーク条件を判定するように構成してもよい。この場合に、図 2 2 の第 1 プログラム例に対するシミュレーション結果及び表示結果を図 2 5 に示す。図 2 2 と比べて「表示結果」欄のみが異なっている。つまり図 2 5 の「表示結果」欄は「シミュレーション結果」欄と同じく当該命令のシミュレーション結果を示している。図 2 5 では、キャンセルされる命令 7 をブレーク条件とした場合に、停止することになる。この場合、ソフトウェア開発者は、命令 7 がキャンセルされるか否かを確認することができる。

【 0 1 4 1 】

また、シミュレーション制御部 3 4 は、命令単位のシミュレーション結果をデバッガ 3 a に出力しているが、これに加えて、パイプラインシミュレーション部 1 0 では命令群をサイクル単位でシミュレーションしているので、命令群単位のシミュレーション結果又は命令群のサイクル単位のシミュレーション結果をデバッガ 3 a に出力するようにしてもよい。また、命令単位のシミュレーション結果と、命令群単位のシミュレーション又は命令群のサイクル単位のシミュレーシヨ

ンとを選択的に切り替える構成としてもよい。

【0 1 4 2】

また、上記実施の形態においてメモリアクセス命令のMEMステージの所要サイクル数は、1サイクル以上の如何なるサイクル数であっても、また、動的に変化する場合であっても、本発明のシミュレーション装置を適用してターゲットプロセッサにおけるサイクル数を正確にシミュレーションすることができる。この場合は、メモリモジュールにおいてメモリアクセスに対する応答（ACK）がどのサイクルで返るかをシミュレーションする構成とすればよい。

【0 1 4 3】

第3レジスタファイルモジュール40は、全レジスタのデータを保持せず、命令により更新されるレジスタのみのデータを保持するようにしてもよい。第4レジスタファイルモジュール41についても同様である。

【0 1 4 4】

【発明の効果】

以上説明してきたように本発明のシミュレーション装置によれば、複数命令を同時実行するプロセッサをターゲットとし、命令単位にシミュレーションを行うことができる。それゆえ、命令群単位のブレークではなく、同時実行される命令単位にブレークすることができる。また、ソフトウェア開発者に対しては、あたかも命令を1つずつ逐次シミュレーションしているかのように見せている。

【0 1 4 5】

しかも、シミュレーション装置は、命令群のサイクル単位のシミュレーションと命令単位のシミュレーションとからなる2段階のシミュレーションを行うので、ターゲットプロセッサのサイクル数を正確にシミュレーションすることができる。

【0 1 4 6】

また、ターゲットプロセッサがフォワーディング機構を有している場合や、遅延命令によりインターロックが発生する場合、キャンセル機構を有している場合でもターゲットプロセッサのサイクル数を正確にシミュレーションすることができる。

【図面の簡単な説明】

【図 1】

本発明の実施の形態におけるシミュレーションシステム 1 の外観を示す図である。

【図 2】

シミュレーション装置 2 においてシミュレーションソフトウェア起動後の表示内容の一例を示す。

【図 3】

デバッグ装置 3 においてデバッグソフトウェア実行中の表示内容の一例を示す。

【図 4】

シミュレーション装置の対象となるターゲットプロセッサの構成を示す図である。

【図 5】

ターゲットプロセッサのパイプラインステージの動作タイミングを示す図である。

【図 6】

パイプライン処理される命令群の一例を示す図である。

【図 7】

パイプラインのサイクル数を説明する図である。

【図 8】

フォワーディングを伴うパイプラインステージの動作タイミングを示す図である。

【図 9】

命令がキャンセルされる場合の他の命令列の例を示す図である。

【図 10】

フォワーディングを伴うパイプライン処理を示す説明図である。

【図 11】

シミュレーション装置の対象となる他のターゲットプロセッサの構成を示す図

である。

【図 1 2】

シミュレーションシステムの構成を示す機能ブロック図である。

【図 1 3】

フェッチ情報の一例を示す図である。

【図 1 4】

デコード情報の一例を示す図である。

【図 1 5】

実行情報の一例を示す図である。

【図 1 6】

完了情報の一例を示す図である。

【図 1 7】

スケジューリングモジュールによる命令群のシミュレーション処理を示すフローチャートである。

【図 1 8】

シミュレーション制御部による命令単位のシミュレーション制御を示すフローチャートを示す。

【図 1 9】

リソース情報変更部の構成を示すブロック図である。

【図 2 0】

通常命令結果生成部の詳細な構成を示すブロック図である。

【図 2 1】

メモリ値退避部の詳細な構成を示すブロック図である。

【図 2 2】

シミュレーション対象の第 1 のプログラム例を示す図である。

【図 2 3】

シミュレーション対象の第 2 のプログラム例を示す図である。

【図 2 4】

シミュレーション対象の第 3 のプログラム例を示す図である。

【図 2 5】

シミュレーション対象の第 4 のプログラム例を示す図である。

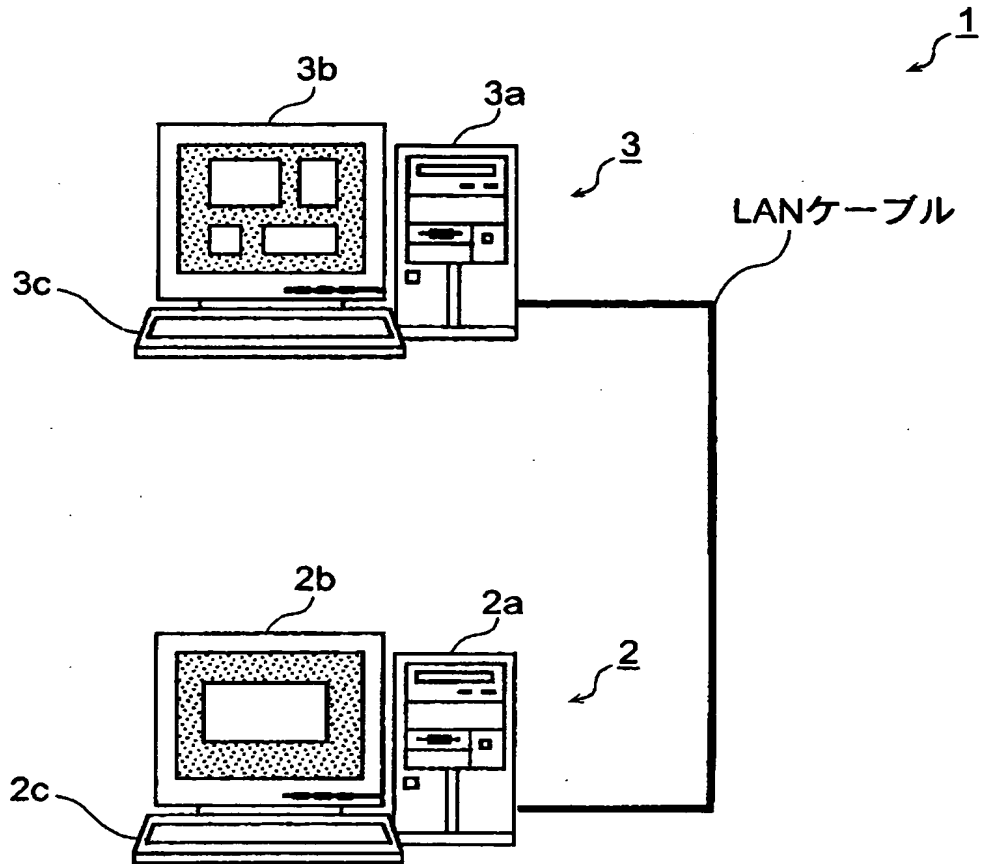
【符号の説明】

- 1 シミュレーションシステム
- 2 シミュレーション装置
 - 2 a 本体装置
 - 2 b 表示装置
 - 2 c 入力装置
- 3 デバッグ装置
 - 3 a デバッガ
 - 3 b 表示装置
 - 3 c 入力装置
- 4 ユーザーインターフェース
- 1 0 パイプラインシミュレーション部
- 1 1 レジスタファイルモジュール
- 1 2 メモリモジュール
- 1 3 共通情報格納部
- 1 4 フェッチモジュール
- 1 5 フェッチ情報格納部
- 1 6 デコードモジュール
- 1 7 デコード情報格納部
- 1 8 実行モジュール
- 1 9 実行情報格納部
- 2 0 完了処理モジュール
- 2 1 完了情報格納部
- 2 2 過去状態更新制御部
- 2 3 スケジューリングモジュール
- 3 0 命令シミュレーション部
- 3 1 レジスタファイルモジュール

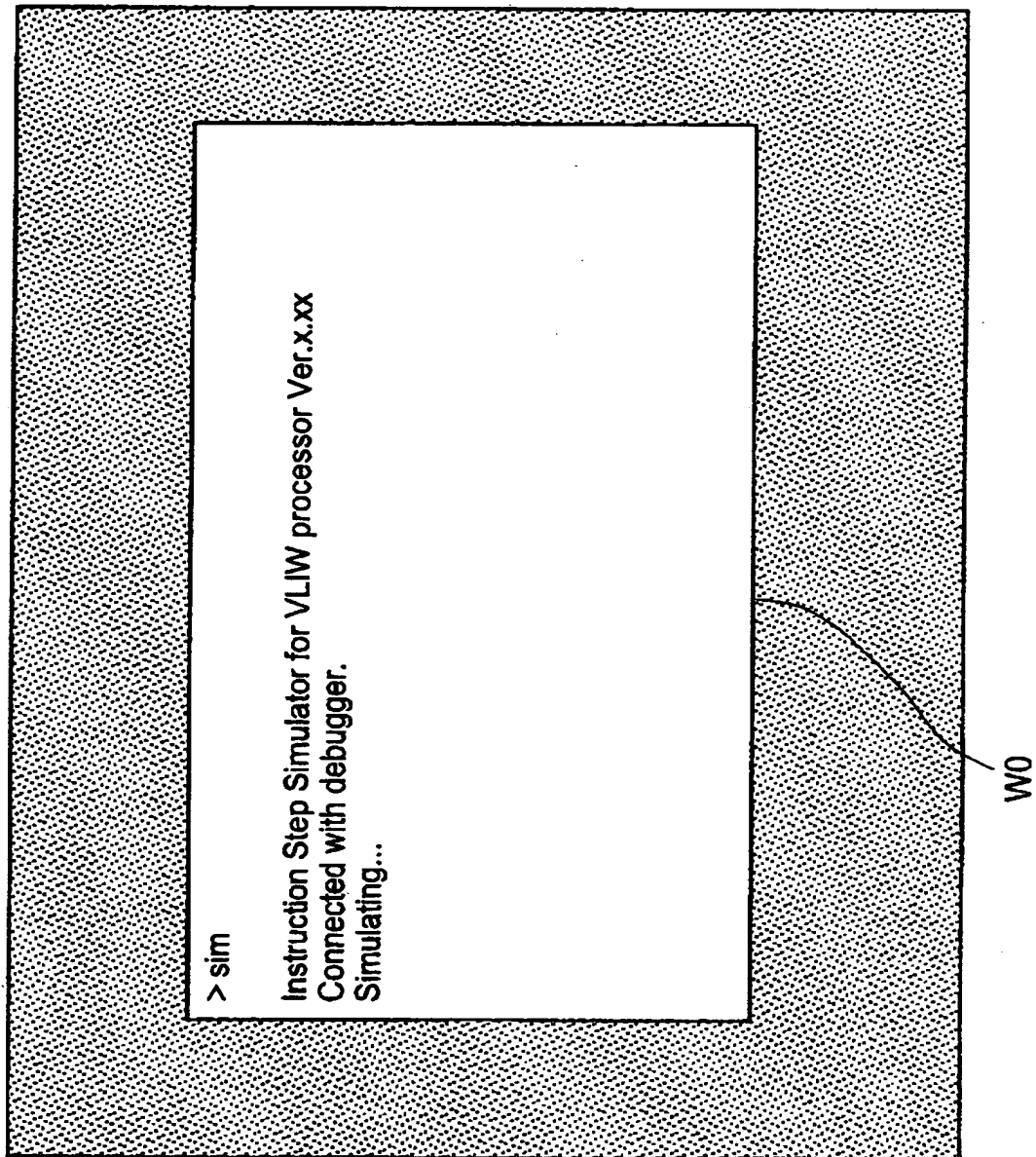
- 3 2 メモリ値退避部
- 3 3 リソース情報変更部
- 3 4 シミュレーション制御部
- 3 4 シミュレーション制御部
- 3 5 通常命令結果生成部
- 3 6 補完部
- 3 7 補完部
- 3 8 メモリ内容選択部
- 3 9 命令毎レジスタファイル退避部
- 4 0 レジスタファイルモジュール
- 4 1 レジスタファイルモジュール
- 4 2 ストア前データ格納部
- 4 3 ストアアドレス格納部
- 4 4 メモリ内容変更部

【書類名】 図面

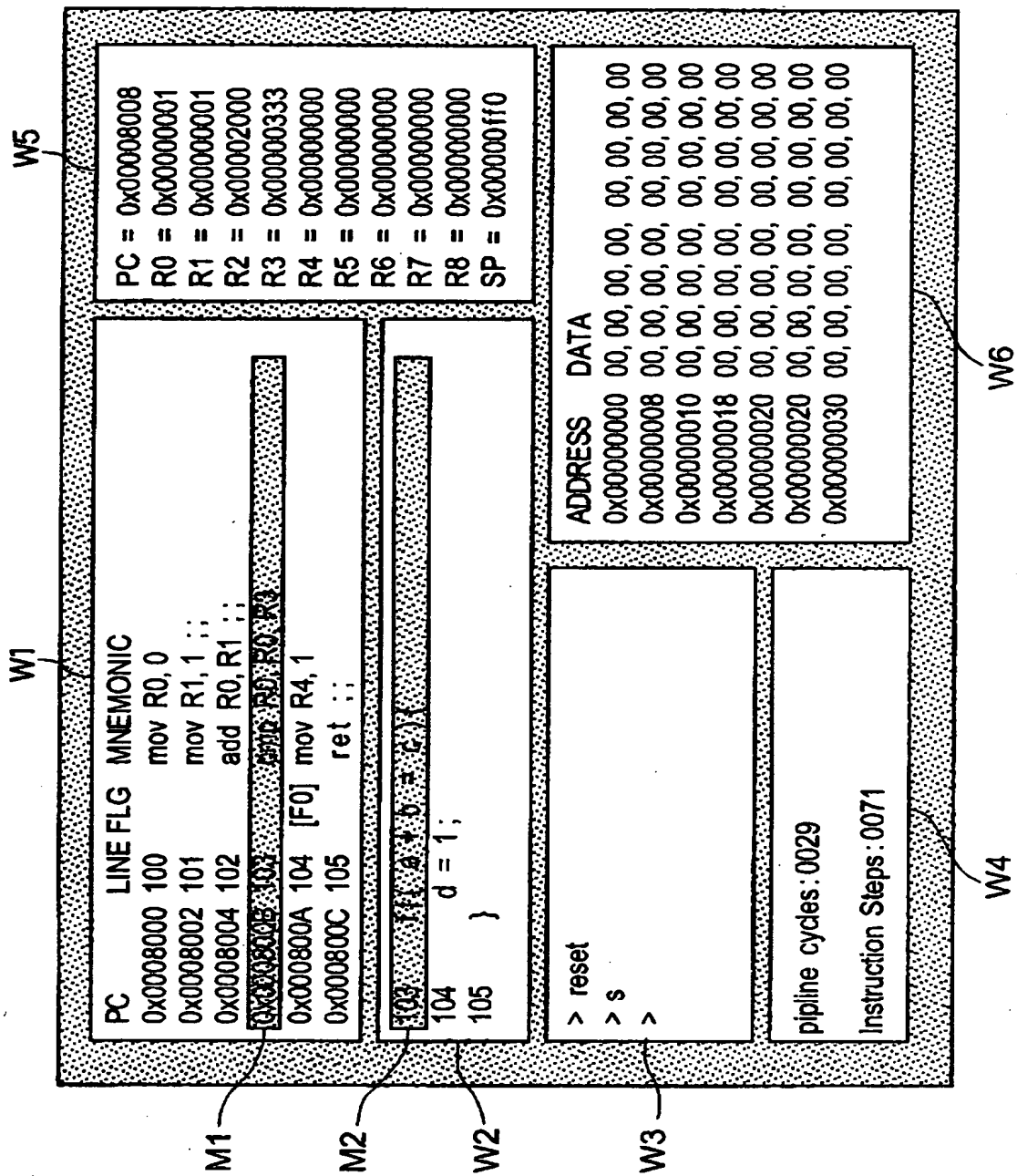
【図 1】



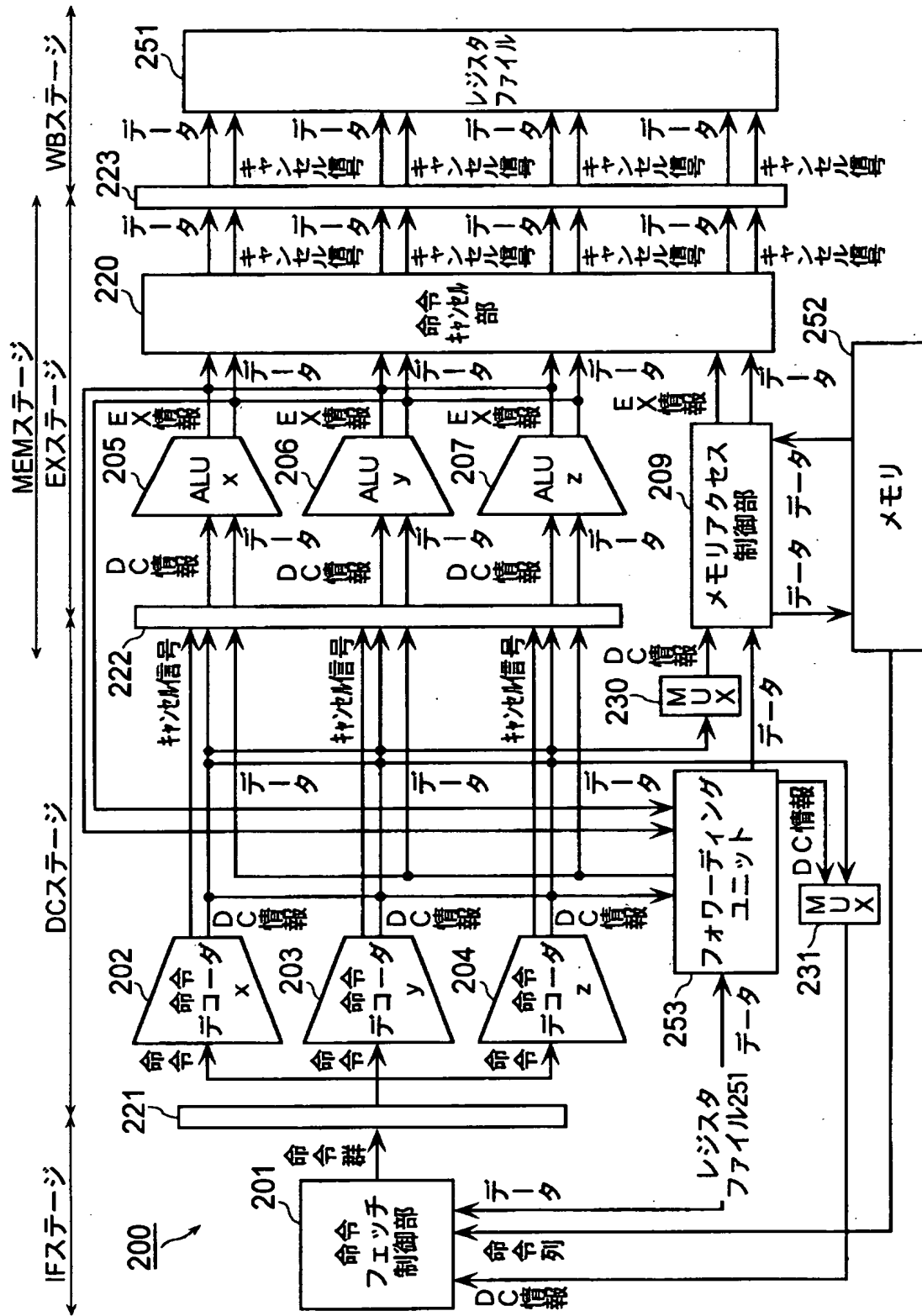
【図 2】



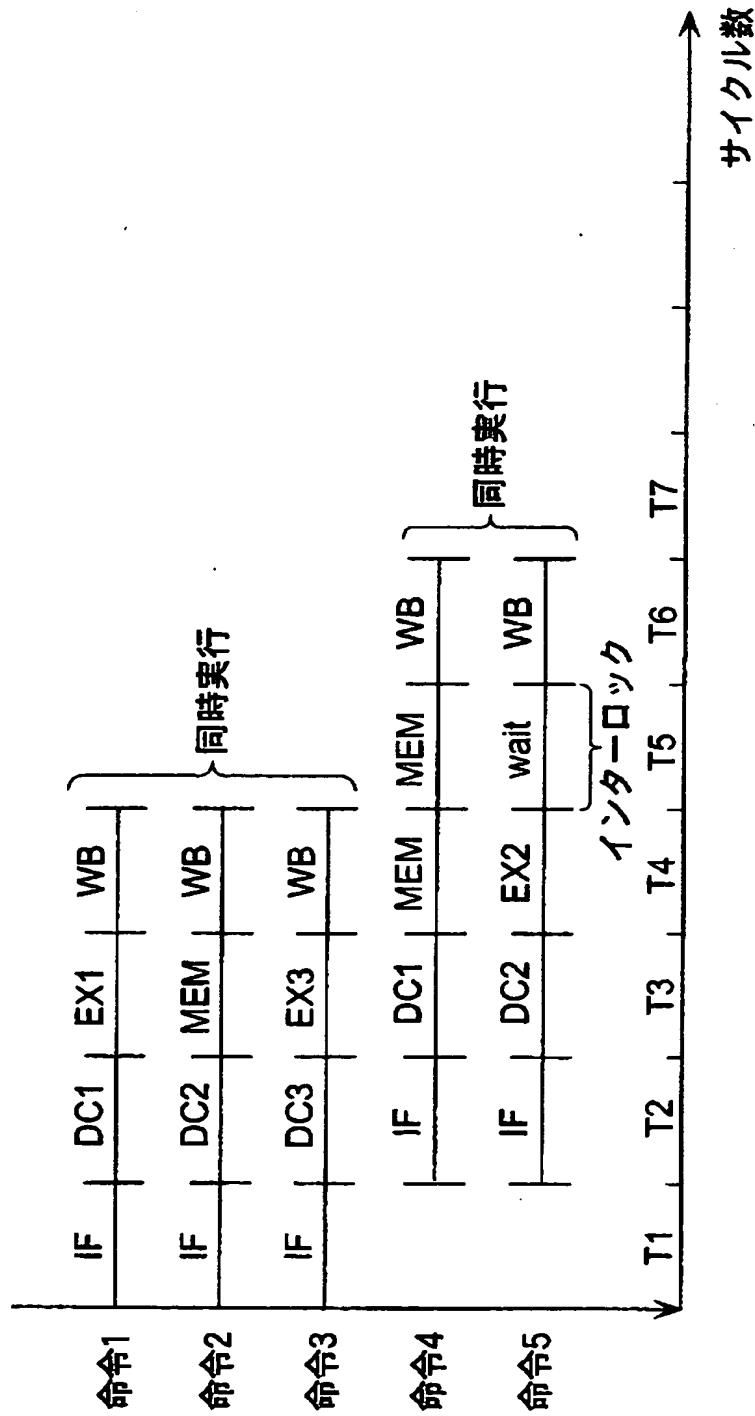
【図 3】



【図 4】



【図 5】



【図 6】

		アドレス	ニーモニック	命令の実行内容	更新リソース
命令群1	命令1	0x8000	sub R0,R1	R0=R0-R1	R0
	命令2	0x8002	add R2,1	R2=R2+1	R2
	命令3	0x8004	ld R3,(R4+)	R3=mem(R4),R4=R4+4	R3,R4
命令群2	命令4	0x8006	st (R4+),R2	mem(R4)=R2,R4=R4+4	mem(R4),R4
	命令5	0x8008	or R5,R6	R5=R5 R6	R5

【図 7】

サイクル	更新されるリソース
N+1	R0,R2,R3,R4
N+2	<なし>
N+3	mem(R4),R4,R5

【図 8】

アドレス		ニーモニック	命令の実行内容
命令6 命令7 命令8	0x9000	cmp F0,R0,R1	R0とR1が等しければF0=1等しくなければF0=0
	0x9002	[F0] add R2,1	F0が1ならばR2=R2+1、F0が1でなければ何もしない。
	0x9004	add R3,1	R3=R3+1

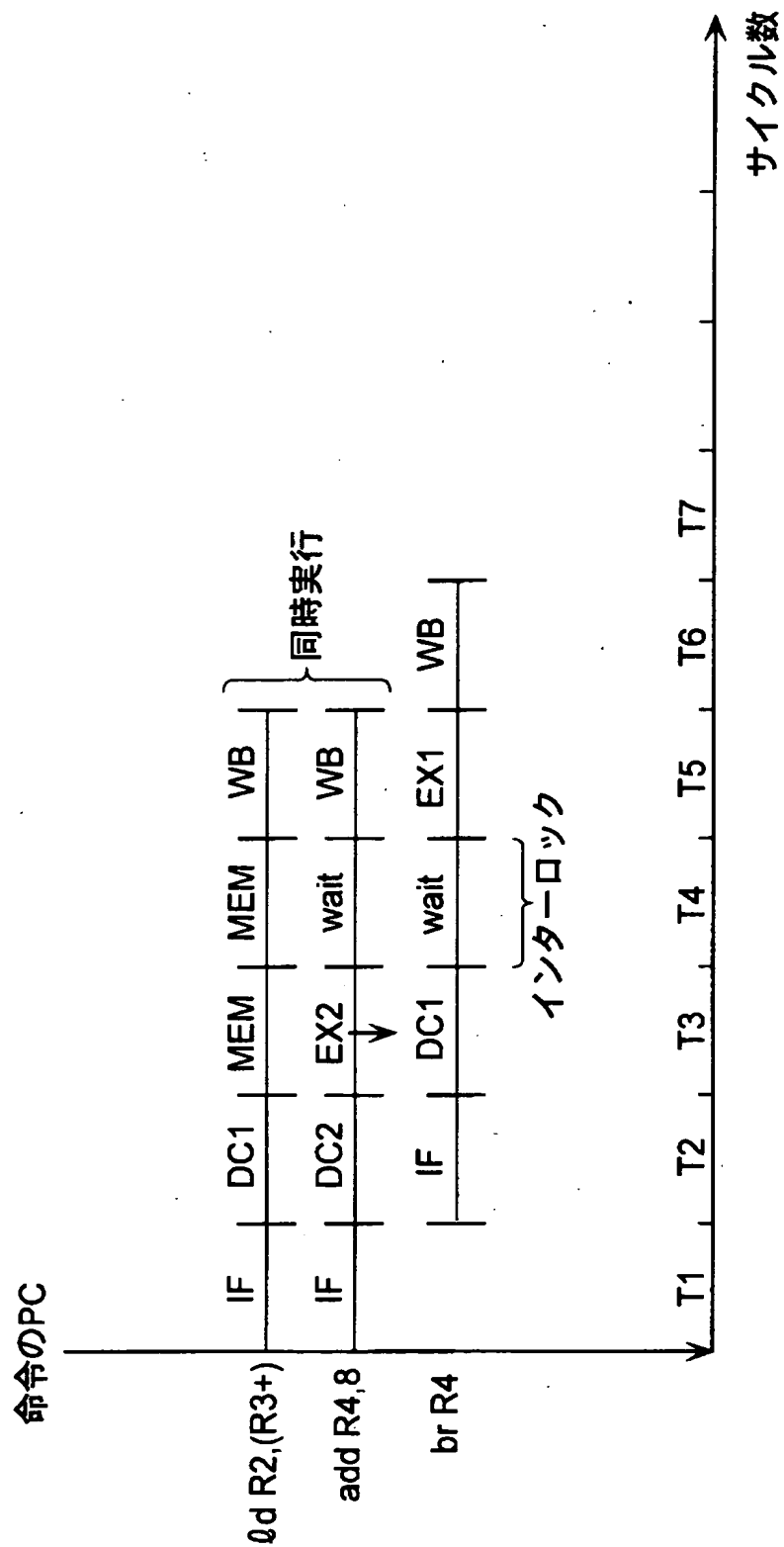
同時実行

【図 9】

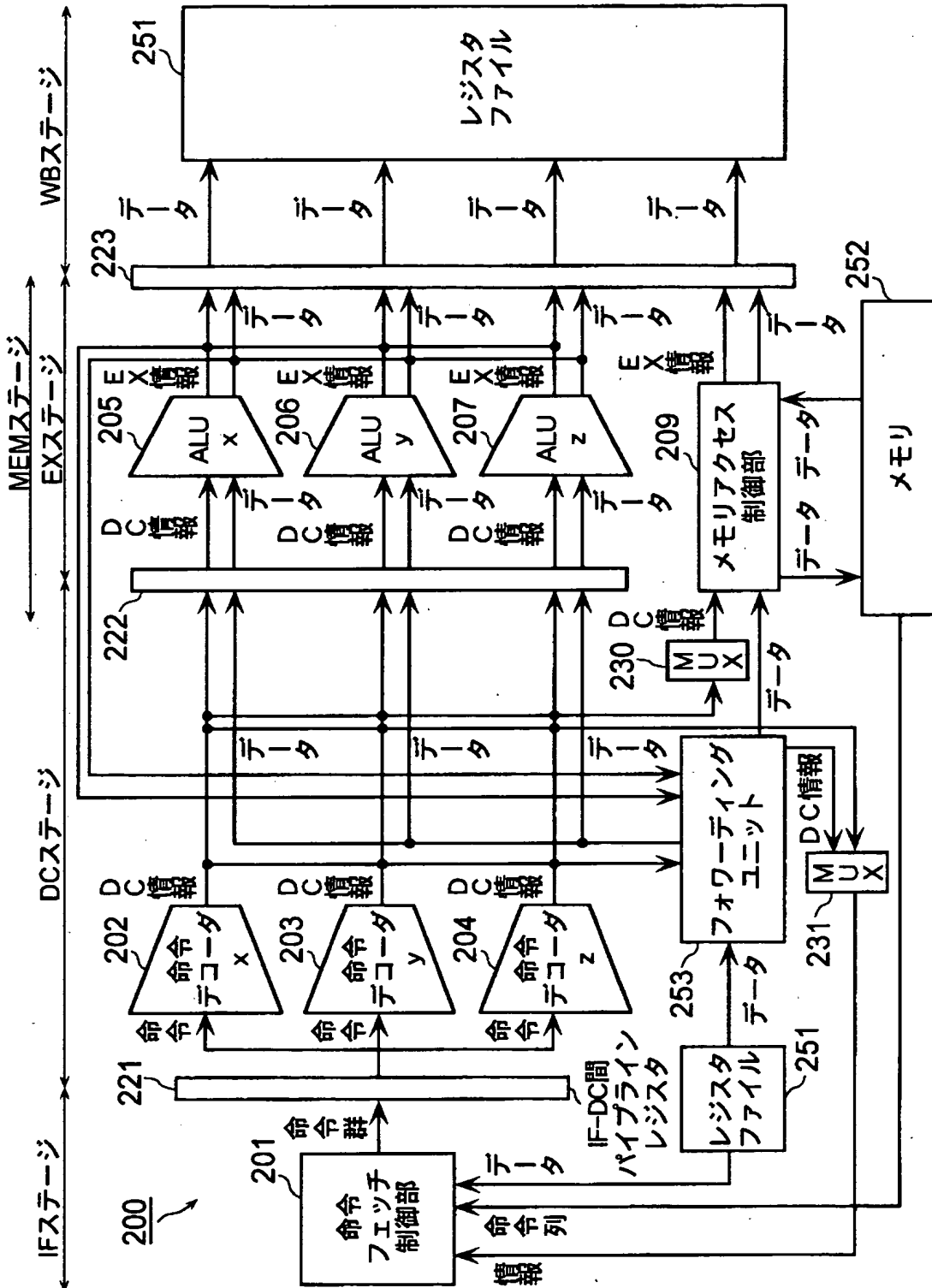
アドレス		ニーモニック	命令の実行内容
命令12 命令13 命令14	0xB000	mov R0,1	R0=1
	0xB002	ld R1,(R2+)	R1=mem(R2),R2=R2+4
	0xB004	mov R1,3	R1=3

同時実行

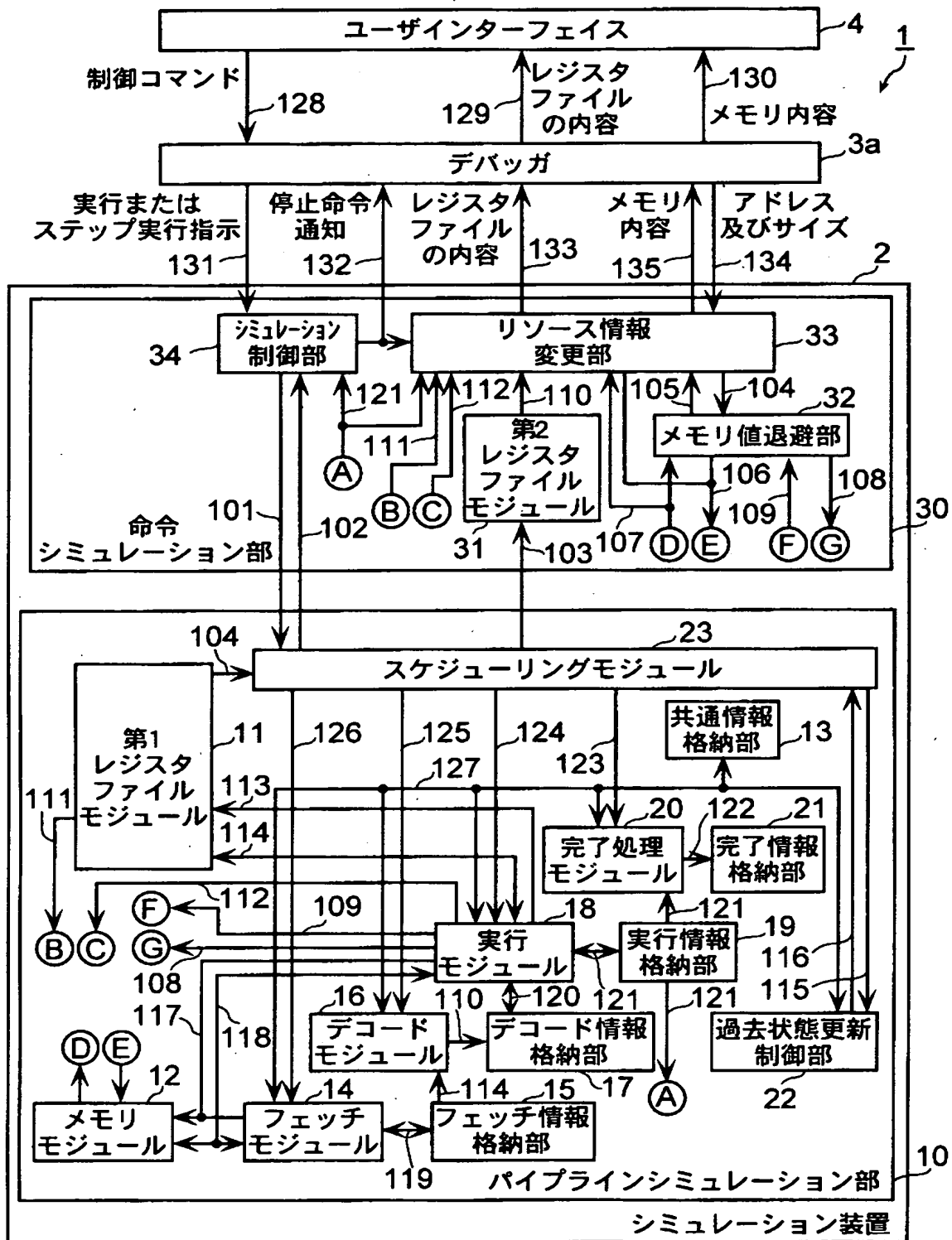
【図 10】



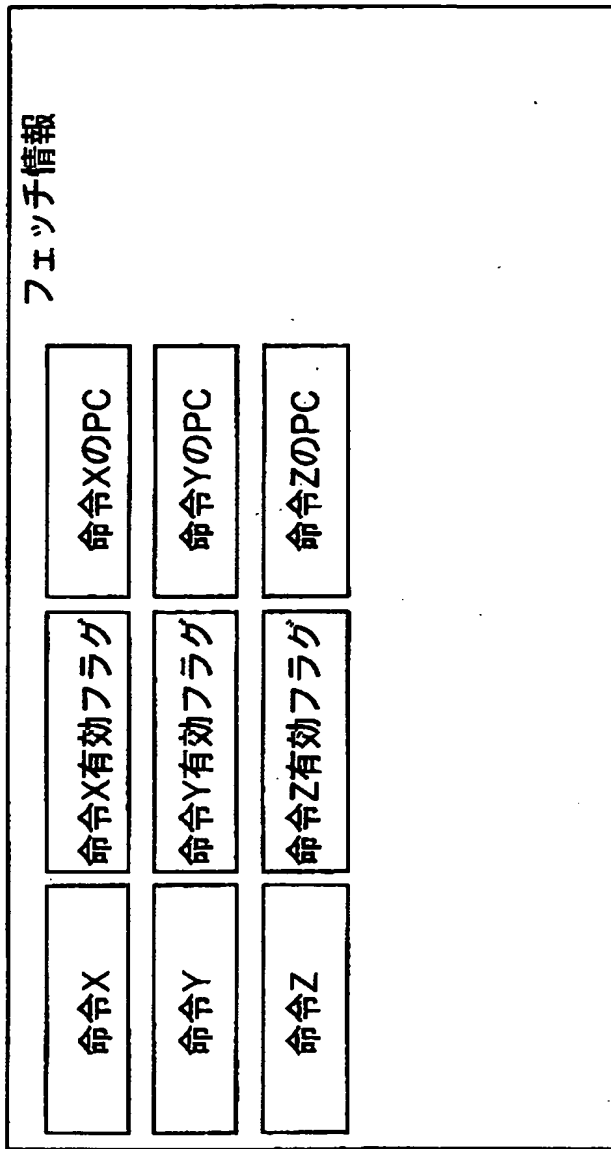
【図11】



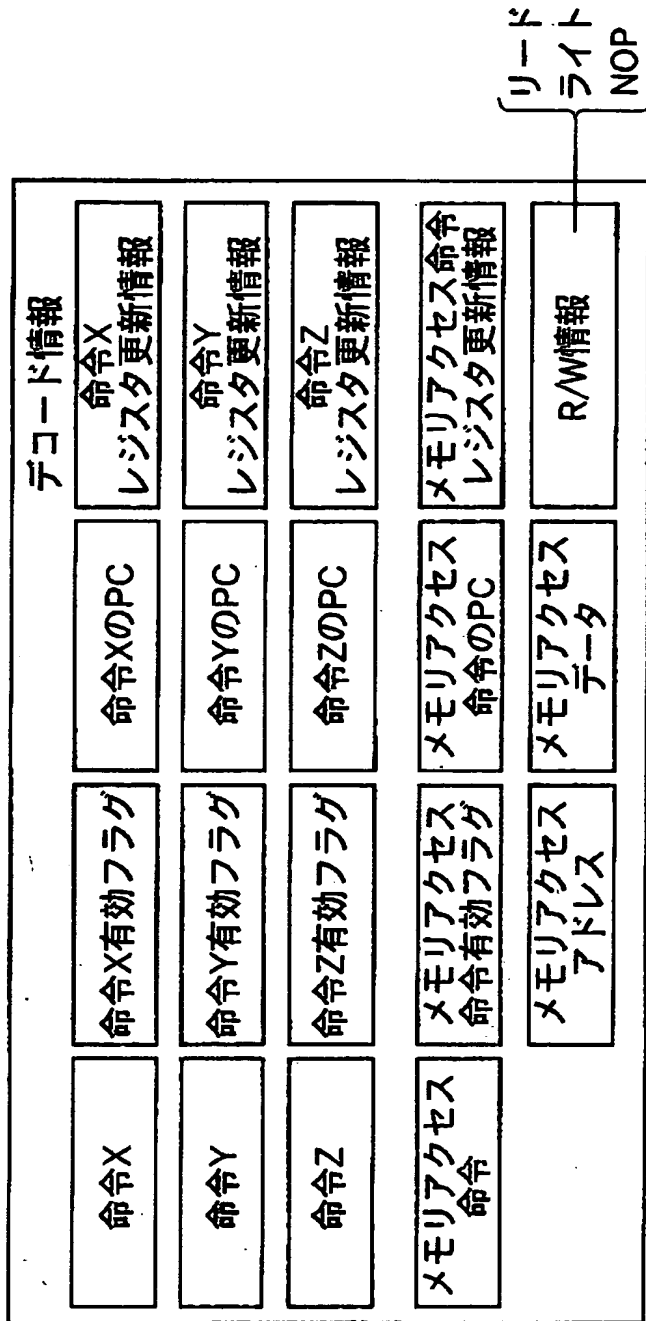
【図 12】



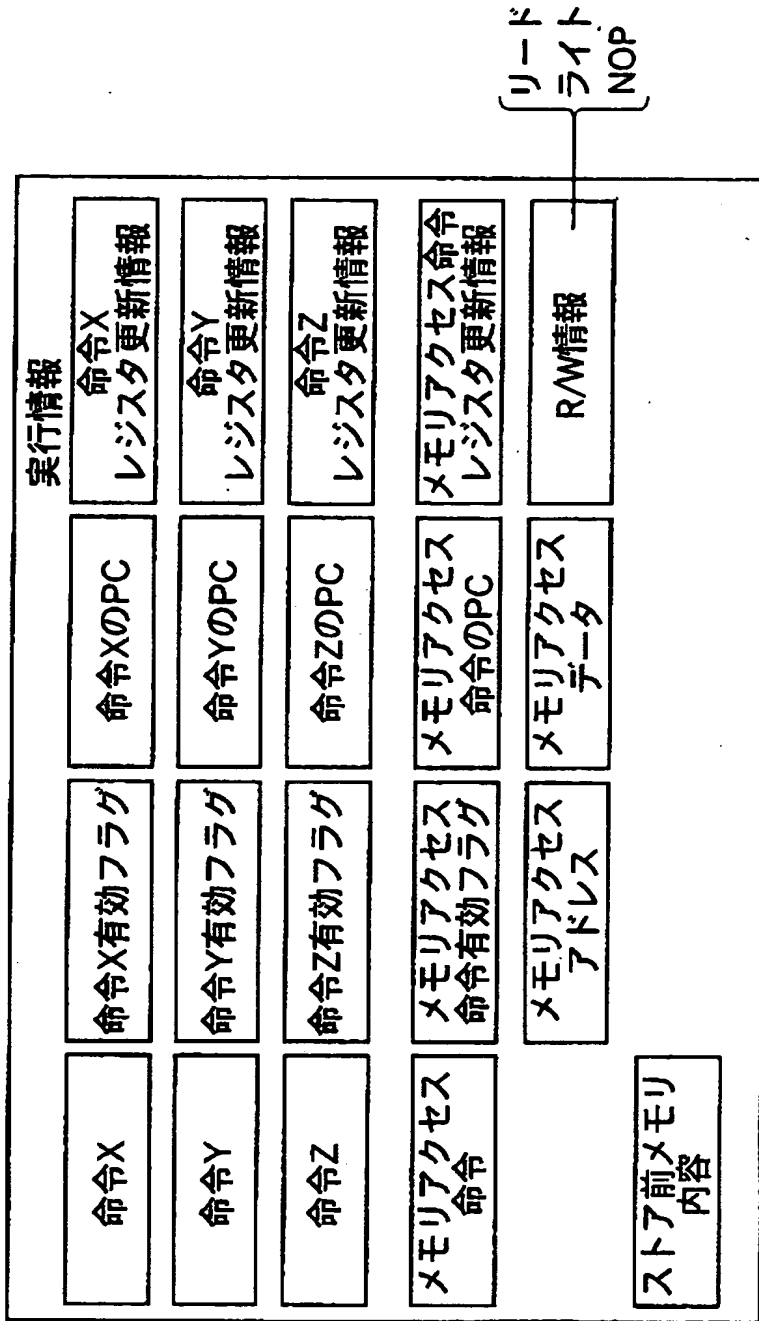
【図 1 3】



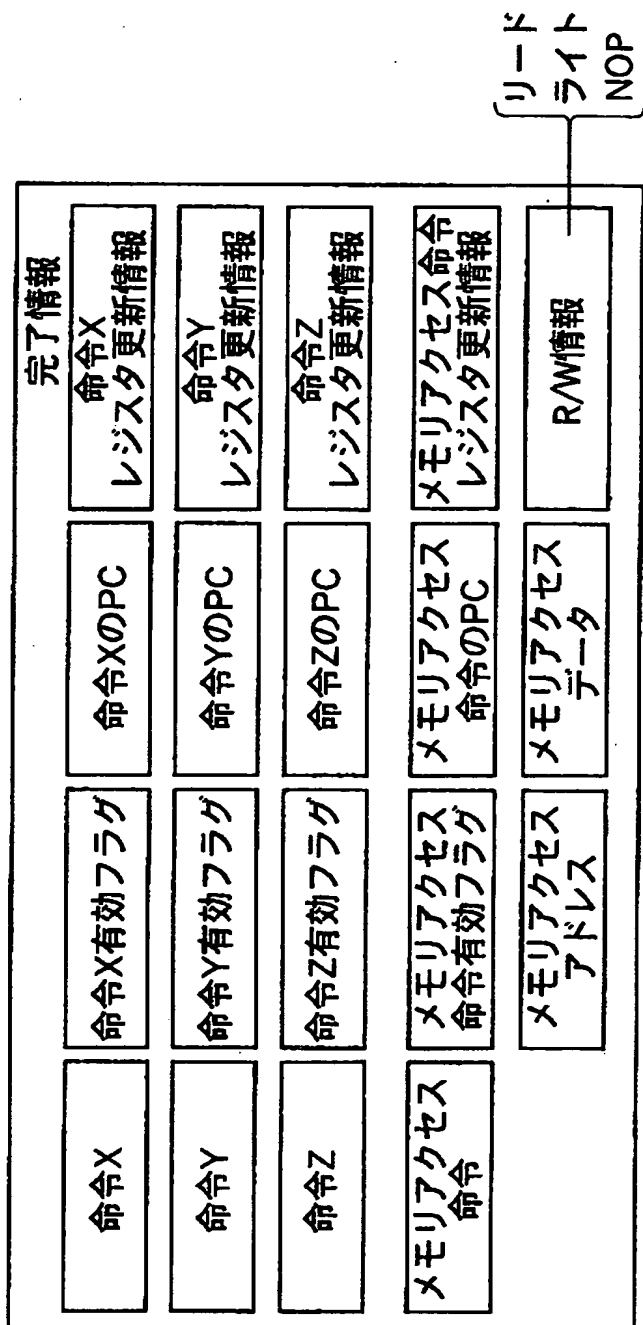
【図 14】



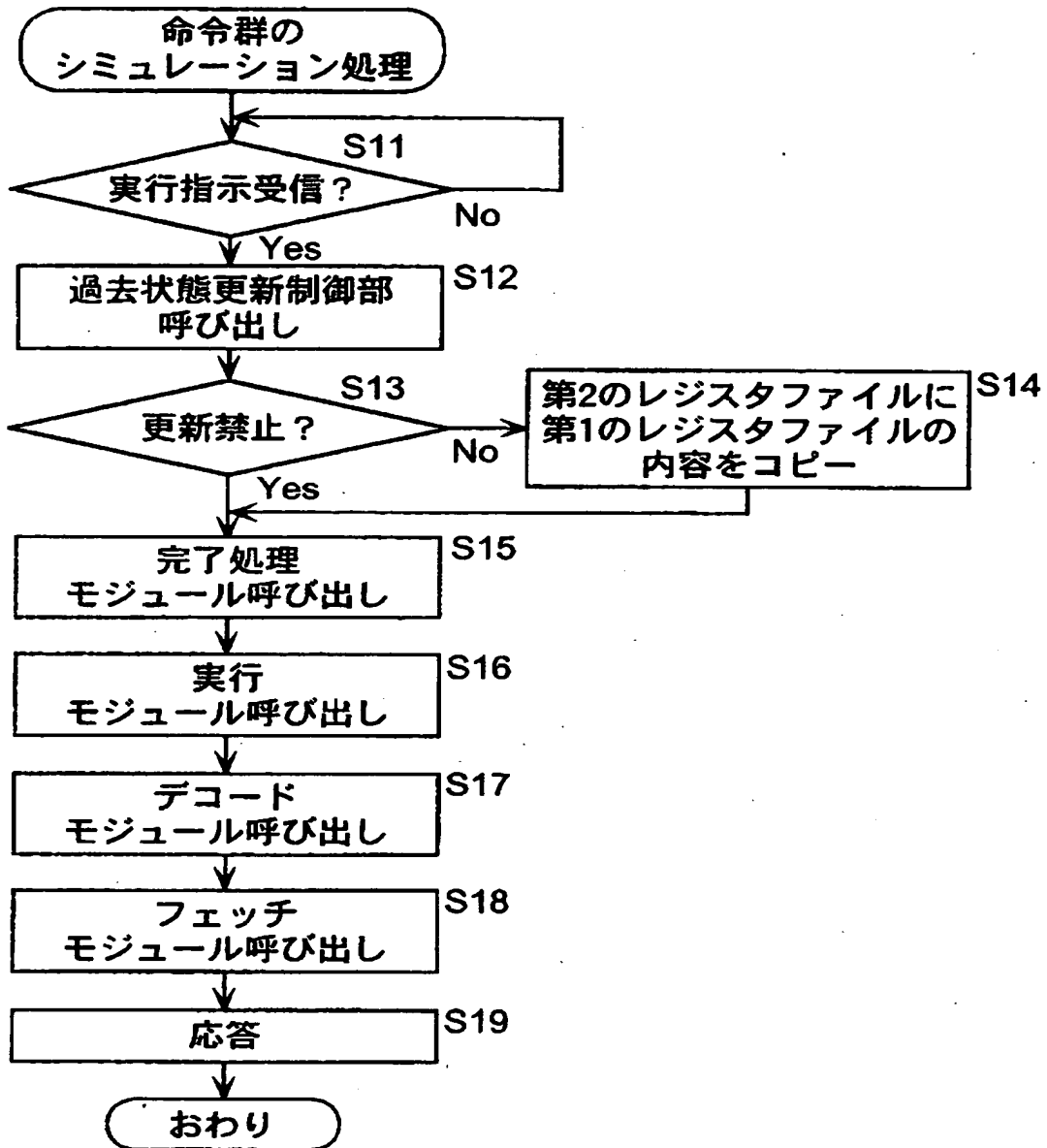
【図 15】



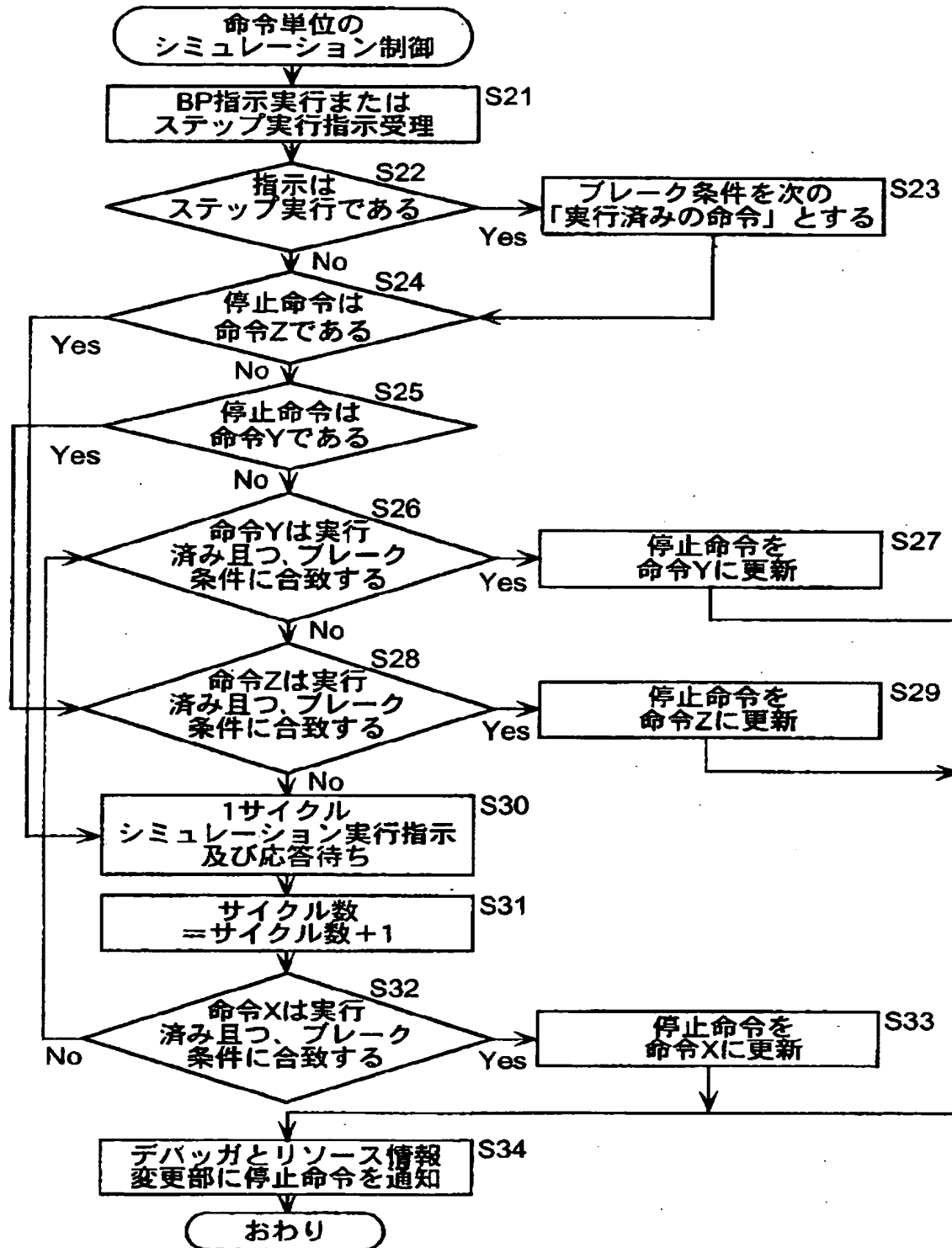
【図 16】



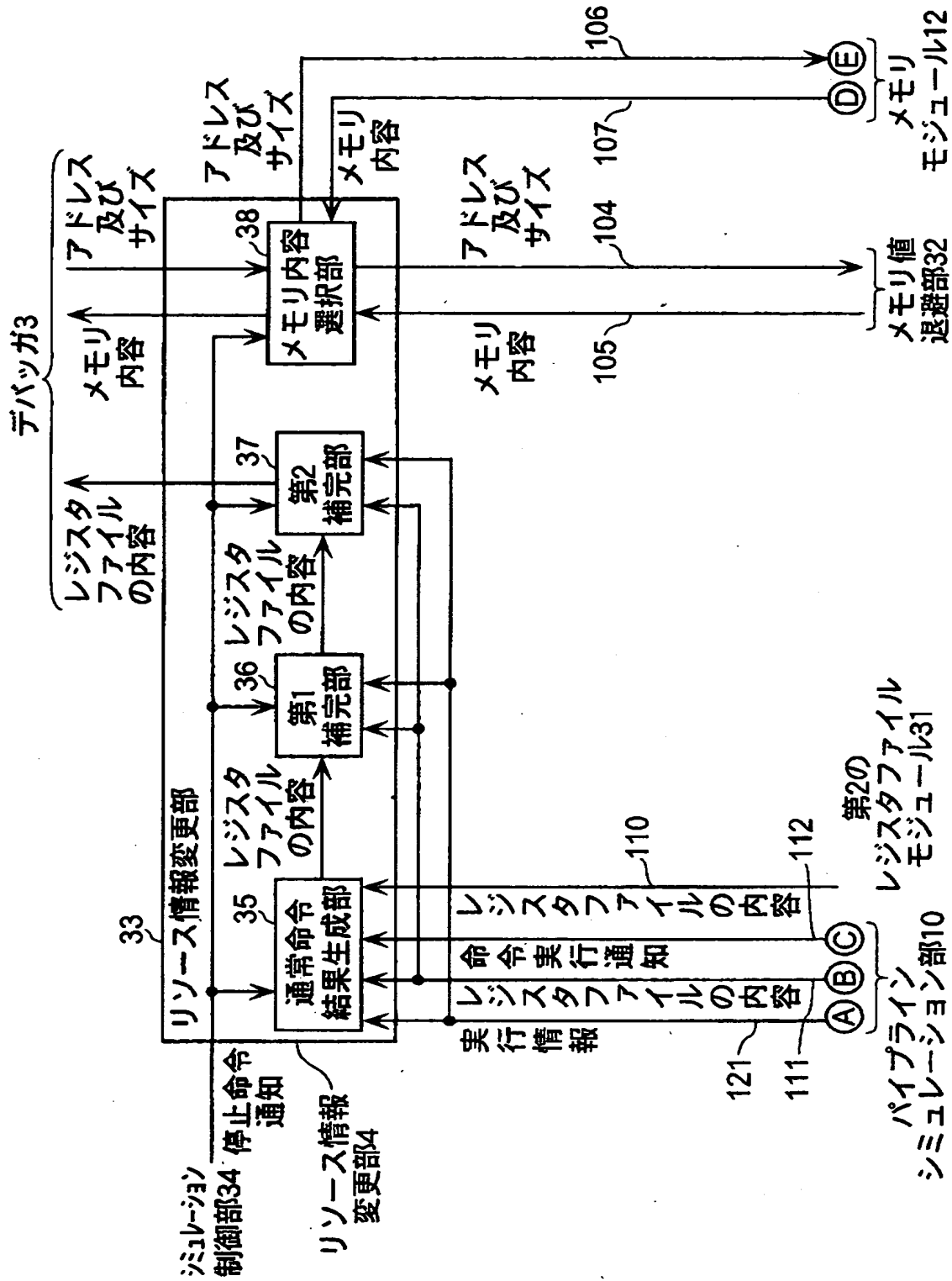
【図 17】



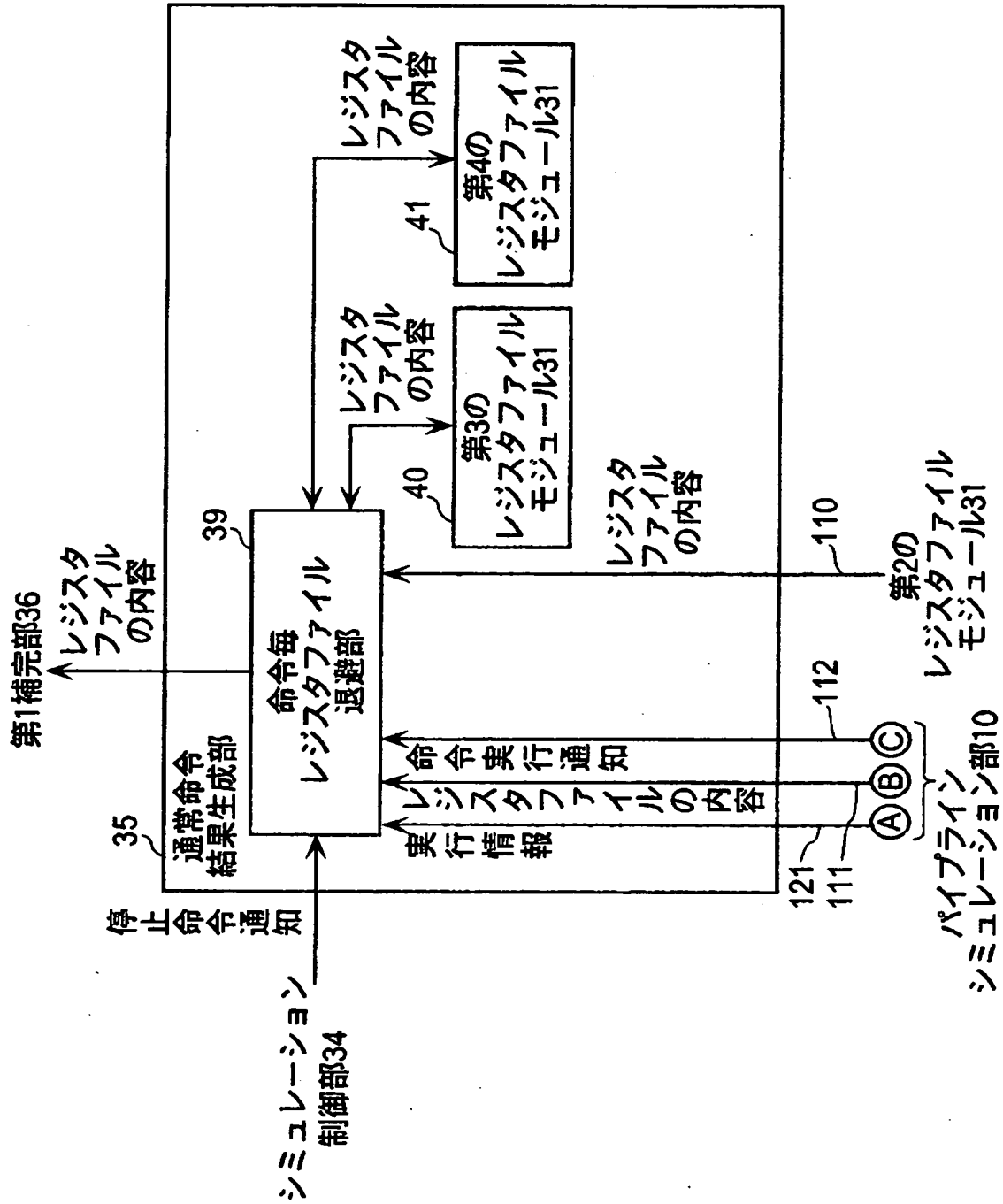
【図 18】



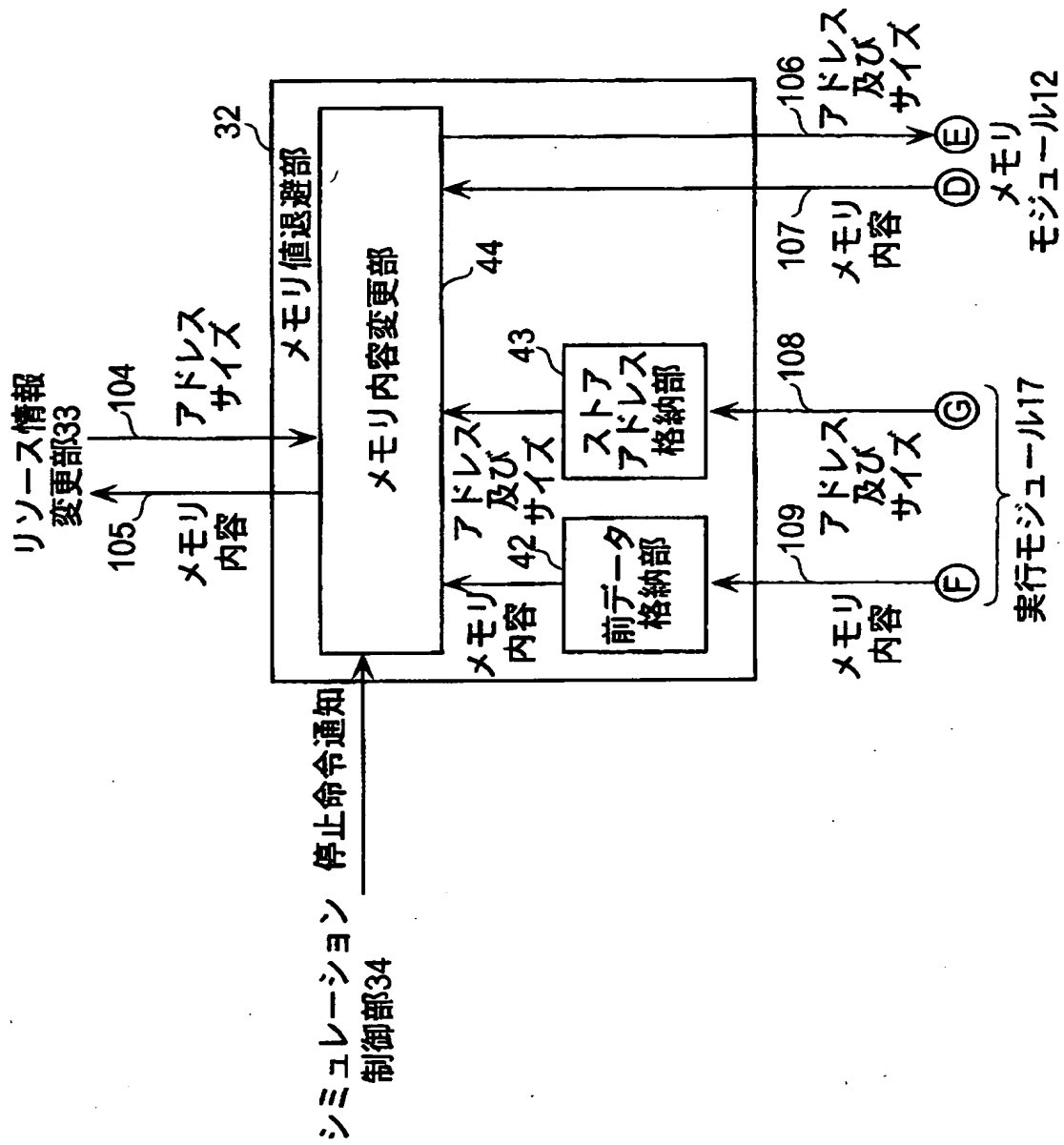
【図19】



【図 20】



【図 21】



【図 2 2】

命令群1	命令6	PC	ニーモニック	シミュレーション結果 {R0,R1,R2,R3,F0}	表示結果 {R0,R1,R2,R3,F0}	停止
命令群1	命令6	0x9000	cmp F0,R0,R1	[1,0,0,0,0]	[1,0,0,0,1]	○
	命令7	0x9002	[F0] add R2,1	[1,0,0,0,0]	[1,0,0,0,0]	×
命令群2	命令8	0x9004	add R3,1	[1,0,0,1,0]	[1,0,0,0,0]	○

【図 2 3】

命令群		PC	ニーモニック	表示結果 {R0,R1,R2,R3,R4,R5,R6}	停止
命令群1	命令1	0x8000	sub_R0,R1	[10,5,0,0,0,1,2]	○
	命令2	0x8002	add R2,1	[5,5,0,0,0,1,2]	○
	命令3	0x8004	ld R3,(R4+)	[5,5,1,0,0,1,2]	○
命令群2	命令4	0x8006	st (R4+),R2	[5,5,1,100,4,1,2]	○
	命令5	0x8008	or R5,R6	[5,5,1,100,8,1,2]	○

【図 2 4】

PC		ニーモニック	表示結果 {R0,R1,R2}	停止
命令群5	命令12	mov R0,1	[0,0,0]	○
	命令13	ld R1,(R2+)	[1,0,0]	○
	命令14	mov R1,3	[1,200,4]	○

第1レジスタ ファイル [R0,R1,R2]	{1,3,4}	第2レジスタ ファイル [R0,R1,R2]	{0,0,0}	第3レジスタ ファイル [R0,R1,R2]	{1,0,0}	第4レジスタ ファイル [R0,R1,R2]	{1,0,4}	メモリ値 退避部	{200}
------------------------------	---------	------------------------------	---------	------------------------------	---------	------------------------------	---------	-------------	-------

【図 2 5】

		PC	ニーモニック	シミュレーション結果 {R0,R1,R2,R3,F0}	表示結果 {R0,R1,R2,R3,F0}	停止
命令群1	命令6	0x9000	cmp F0, R0, R1	[1,0,0,0,0]	[1,0,0,0,0]	○
	命令7	0x9002	[F0] add R2, 1	[1,0,0,0,0]	[1,0,0,0,0]	○
命令群2	命令8	0x9004	add R3, 1	[1,0,0,1,0]	[1,0,0,1,0]	○

【書類名】 要約書

【要約】

【課題】 複数命令を同時実行するプロセッサを対象とし、命令単位の実行過程がわかるようにシミュレーションを行うシミュレーション装置を提供する。

【解決手段】 シミュレーション装置 2 においてパイプラインシミュレーション部 1 0 は、同時実行されるべき複数命令からなる命令群をシミュレーションし、命令シミュレーション部 3 0 は、パイプラインシミュレーション部 1 0 によるシミュレーション結果に基づいて、前記命令群中の命令毎のシミュレーション結果を生成する。

【選択図】 図 1 2

認定・付加情報

特許出願の番号	特願2002-360362
受付番号	50201881426
書類名	特許願
担当官	第七担当上席 0096
作成日	平成14年12月13日

<認定情報・付加情報>

【提出日】 平成14年12月12日

出 願 人 履 歴 情 報

識別番号 [000005821]

1. 変更年月日 1990年 8月28日

[変更理由] 新規登録

住 所 大阪府門真市大字門真1006番地
氏 名 松下電器産業株式会社